Quantum Development: Early Best Practices *from QML*

Innovation with Quantum Computing?!

GSE Architecture & Innovation Working Group

15 January 2020

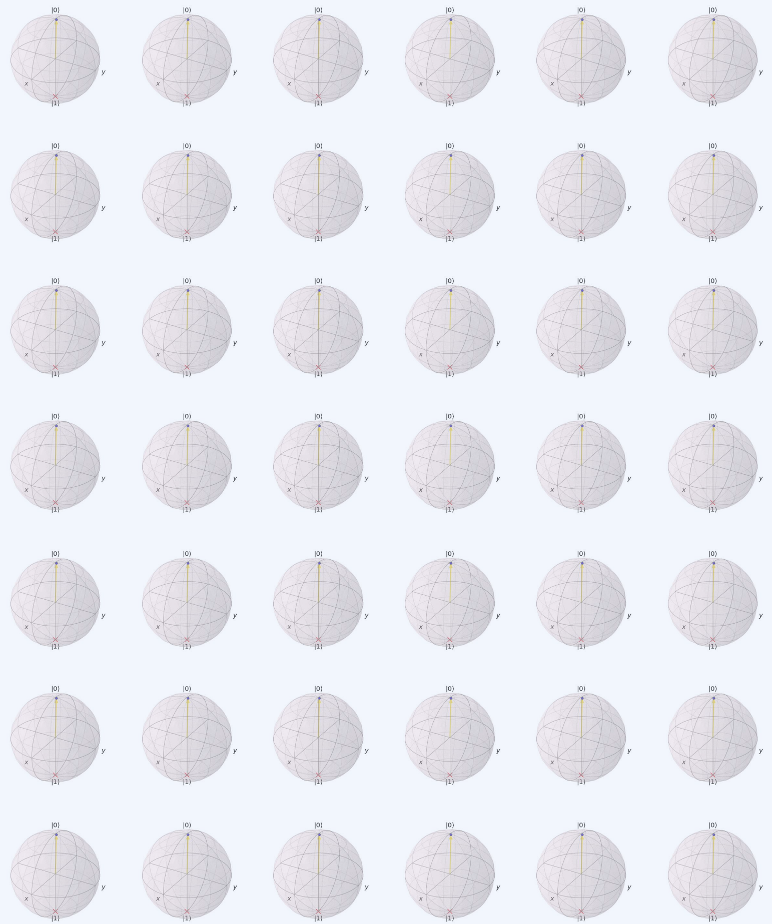
Karel Dumon

Co-founder & Software Lead at Miraex

@kareldumon

MIRAEX

GSE GUIDE SHARE EUROPE



Karel Dumon - Co-founder & Software/ML lead at **Miraex**

background: Engineering Physics *(UGent)*, Machine Learning *(ML6)*

Qiskit advocate, quantum computing hackathons (CDL, IBM Qiskit Camp), meetup

MIRAEX

Next-gen sensing, networking and computing

Team of experts in quantum optics, computing and machine learning (spin-off EPFL)

- Photonics **sensors** >> access data that no one else can (extreme environments)
- Quantum photonics **transducers** >> access to extremely low energy signals (quantum level)
- Quantum computer **interconnect** >> enabling quantum computers networking
- **Software platform** (data & ML) >> data access & analysis

MIRAEX

QUANTUM COMPUTING Qiskit Advocate IBM

CREATIVE DESTRUCTION LAB

EPFL

GHENT UNIVERSITY

Talk outline

Quantum algorithms landscape: how to push forward?

Quantum machine learning: what's in a name?

Example projects & tooling:

- Using machine learning for quantum: PennyLane
- Putting quantum in machine learning: Qiskit & PyTorch

Distilling some 'early development experiences/best practices'

FEASIBILITY

SPEEDUP

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)		
Small speedup (polynomial)		
Unknown speedup		

Chemistry

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)		Hamiltonian simulation
Small speedup (polynomial)		
Unknown speedup	VQE	

Chemistry Optimization

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)		Hamiltonian simulation
Small speedup (polynomial)		Grover's search & generalization (NP-hard optimization, Monte Carlo,...)
Unknown speedup	VQE QAOA, noisy adiabatic	

Chemistry Optimization **Machine Learning**

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)		Hamiltonian simulation **HHL-based algos (~qBLAS)**
Small speedup (polynomial)		Grover's search & generalization (NP-hard optimization, Monte Carlo,...)
Unknown speedup	VQE QAOA, noisy adiabatic **(quantum) neural nets**	**Tensor networks**

Chemistry Optimization **Machine Learning** Cryptography

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)	Quantum Key Distribution	Hamiltonian simulation **HHL-based algos (~qBLAS)** Breaking Cryptosystems
Small speedup (polynomial)		Grover's search & generalization (NP-hard optimization, Monte Carlo,...)
Unknown speedup	VQE QAOA, noisy adiabatic **Neural nets**	**Tensor networks**

Chemistry Optimization **Machine Learning** Cryptography *Quantum Supremacy*

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)	Quantum Key Distribution *Quantum Supremacy & Certifiable Randomness*	Hamiltonian simulation **HHL-based algos (~qBLAS)** Breaking Cryptosystems
Small speedup (polynomial)		Grover's search & generalization (NP-hard optimization, Monte Carlo,...)
Unknown speedup	VQE QAOA, noisy adiabatic **Neural nets**	**Tensor networks**

Chemistry Optimization **Machine Learning** Cryptography *Quantum Supremacy*

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)	Quantum Key Distribution *Quantum Supremacy & Certifiable Randomness*	Hamiltonian simulation **HHL-based algos (~qBLAS)** Breaking Cryptosystems
Small speedup (polynomial)		Grover's search & generalization (NP-hard optimization, Monte Carlo,...)
Unknown speedup	VQE QAOA, noisy adiabatic **Neural nets**	**Tensor networks**

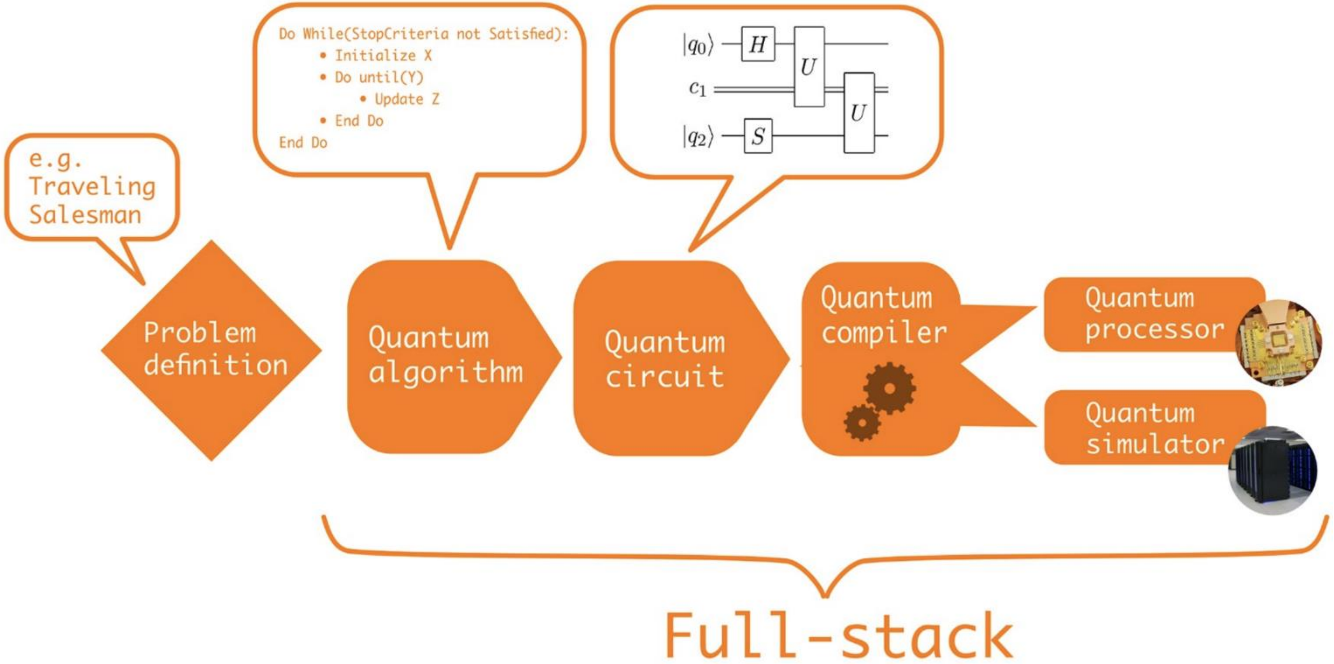
Quantum sensing, Quantum Internet, ...

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)	**where we want to be!**	**most algorithms**
Small speedup (polynomial)		**most algorithms**
Unknown speedup	**most algorithms**	

	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)	**where we want to be!**	**most algorithms**
Small speedup (polynomial)		**most algorithms**
Unknown speedup	**most algorithms**	

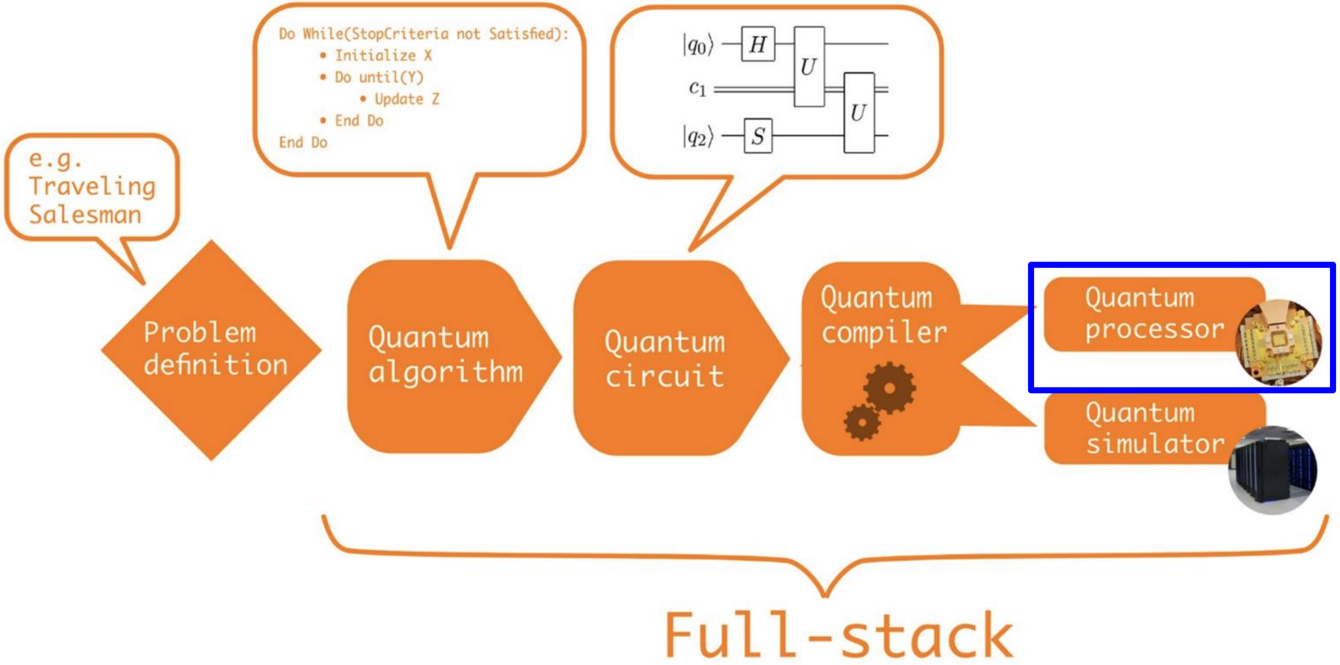
HOW?

Algorithm on a gate-model quantum computer



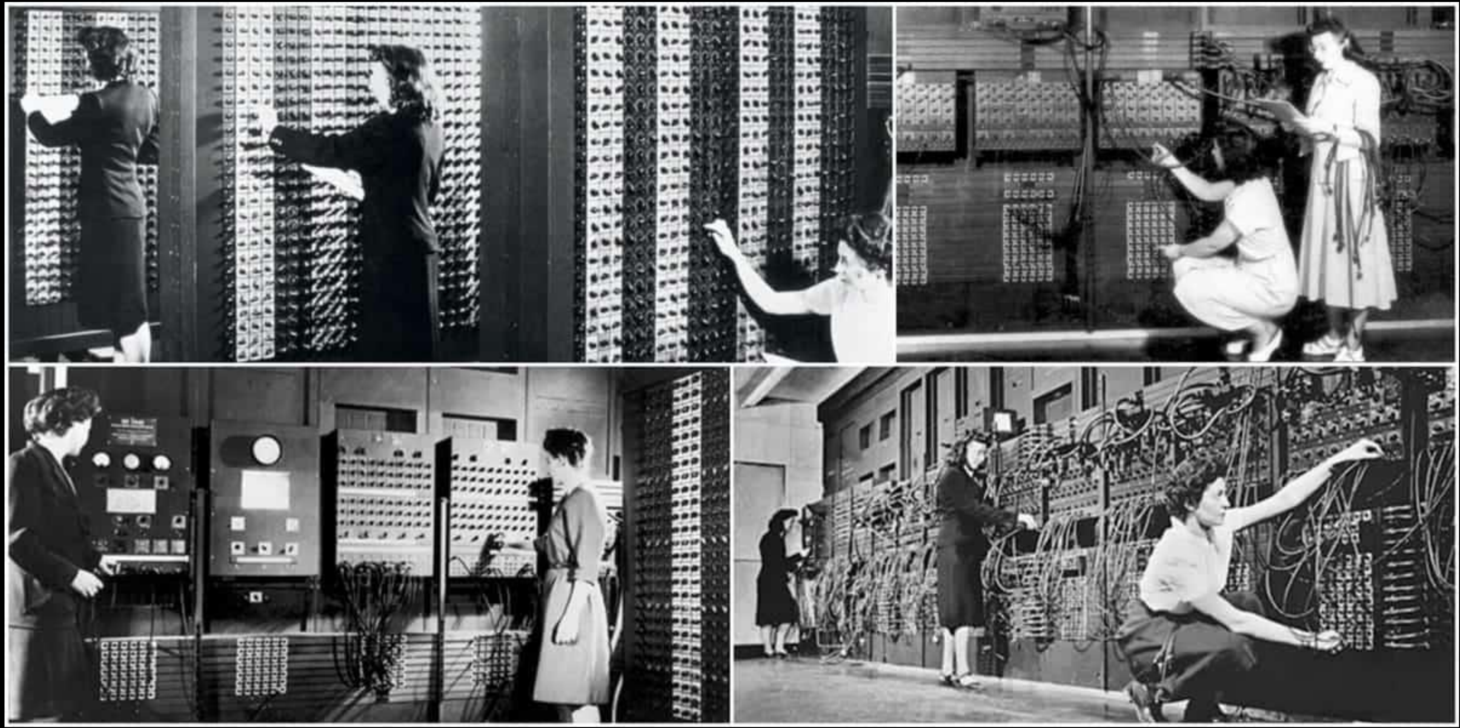
M. Fingerhuth, T. Babej, P. Wittek. (2018). Open source software in quantum computing. *PLOS ONE* 13(12):e0208561.

Hardware

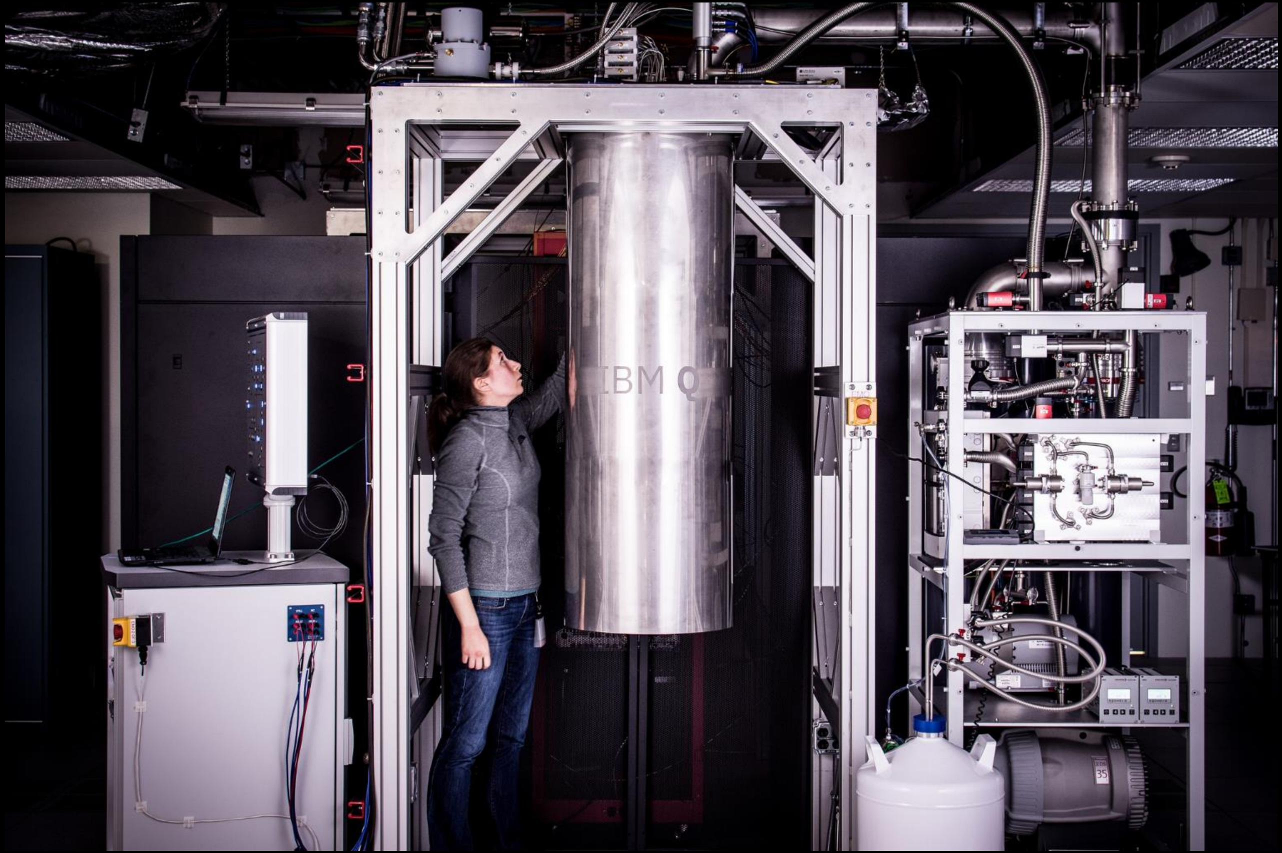


 M. Fingerhuth, T. Babej, P. Wittek. (2018). Open source software in quantum computing. *PLOS ONE* 13(12):e0208561.

Exploiting NISQ hardware



Exploiting NISQ hardware



DiVincenzo criteria

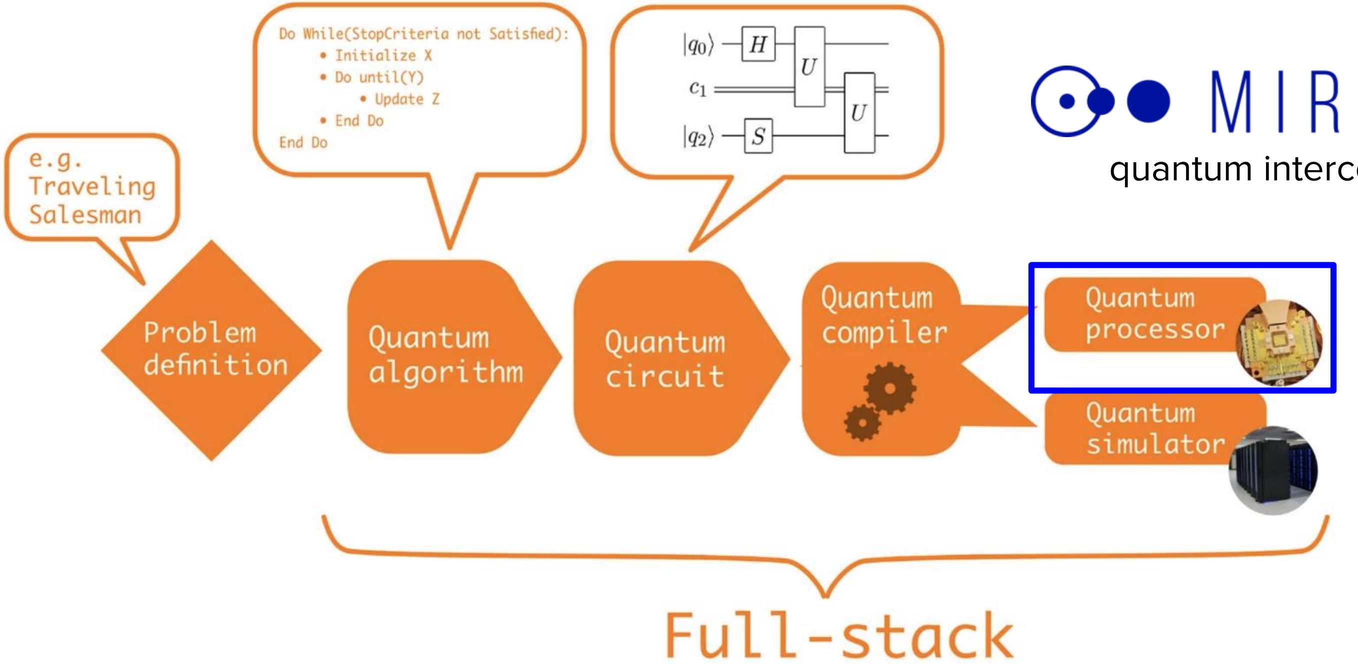
Necessary for quantum computation:

1. A scalable physical system with well characterized qubit
2. The ability to initialize the state of the qubits to a simple fiducial state
3. Long relevant decoherence times
4. A "universal" set of quantum gates
5. A qubit-specific measurement capability

Necessary for quantum communication:

1. The ability to interconvert stationary and flying qubits
2. The ability to faithfully transmit flying qubits between specified locations

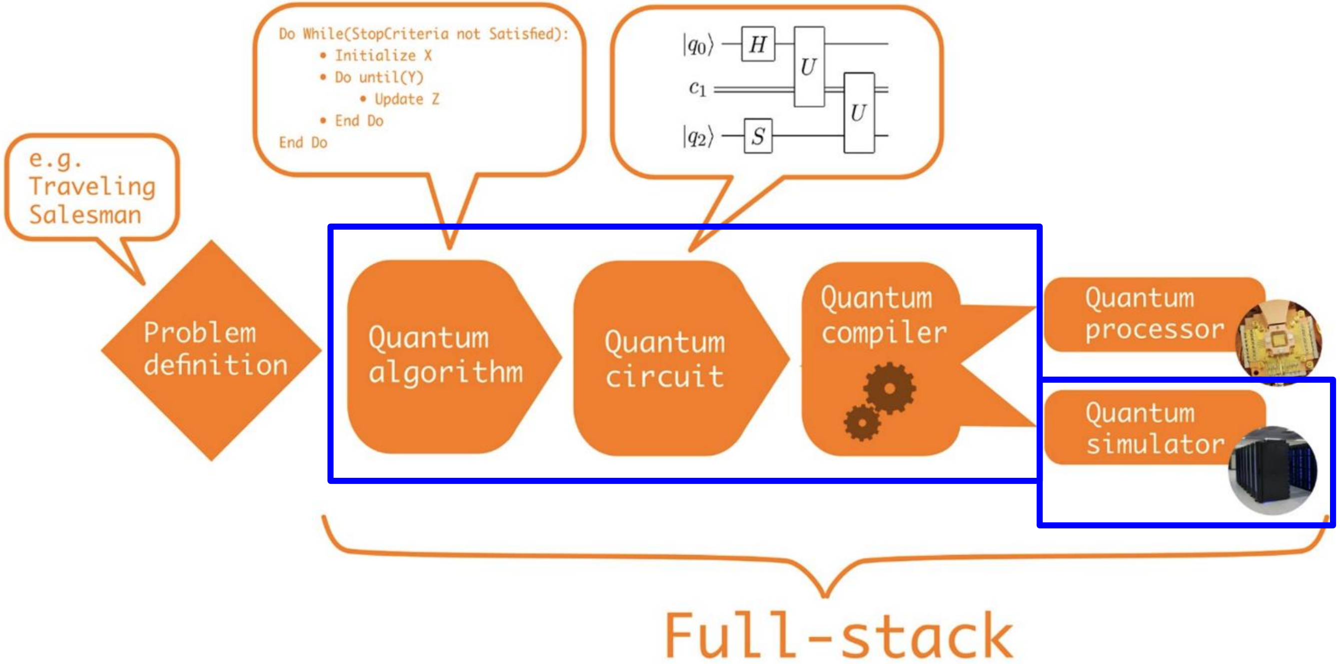
Hardware



MIRAEX quantum interconnects

M. Fingerhuth, T. Babej, P. Wittek. (2018). Open source software in quantum computing. *PLOS ONE* 13(12):e0208561.

Software



M. Fingerhuth, T. Babej, P. Wittek. (2018). Open source software in quantum computing. *PLOS ONE* 13(12):e0208561.

HOME MANIFESTO LEARN CODE EVALUATION

Fork me o

We're organizing the quantum open source track at FOSDEM on February 1-2 2020! Stay on top of newest developments in the quantum open source space, meet with developers, contributors and maintainers of projects and contribute to the growing ecosystem at the quantum computing workshop. **Read more.**

Quantum Open Source Foundation

Supporting the development and standardization of open tools for quantum computing.

Become a supporter

Follow us on GitLab

Follow us on GitHub

THE TEAM →

Find out more about the team behind the Quantum Open Source Foundation (QOSF).

HOME MANIFESTO LEARN CODE EVALUATION

We're organizing the quantum open source track at FOSDEM on February 1-2 2020! Stay on
source space, meet with developers, contributors and maintainers of projects and contribut
computing workshop. **Read more.**

Quantum Open Source Foundation

Supporting the development and standardization of open tools for quantum computing.

Become a supporter

Follow us on GitLab

Follow us on GitHub

- Quantum full-stack libraries
- Quantum simulators
- Quantum annealing
- Quantum algorithms
- Quantum compilers
- Quantum assembly
- Quantum cryptography
- Experimental quantum computing
- Quantum fun
- Quantum tools
- Abandoned projects

Quantum simulators

- BLACK-STONE
- Cliffords.jl
- Liqui|>
- PIQS
- QCGPU
- QCL
- QOCS
- QSimulator.jl
- QTop
- QWIRE
- QImp
- Qrack
- QuEST
- QuSim
- QuTiP
- QuaC
- Quantum Circuit

Quantum full-stack libraries

- Cirq
- Forest
- Ocean
- ProjectQ
- Q#
- Qiskit
- Qiskit-JS
- Strawberry Fields
- XACC

Quantum algorithms

- Adapt
- FermiLib
- Grove
- OpenFermion
- PennyLane
- QFog
- Qiskit Aqua
- Qiskit Tutorial
- Quantum Katas
- QuantumFlow
- QuantumTomography.jl
- Quantum_Edward
- XACC Examples
- XACC QChem
- XACC VQE

Quantum fun

- Entanglion
- QSEL
- Quantum Awesomeness
- Quantum Battleships
- Quantum Catsweeper
- Quantum Music Composer for IBM Q
- Quantum Music Composer for Rigetti

Quantum simulators

- BLACK-STONE
- Cliffords.jl
- Liqui|>
- PIQS
- QCGPU
- QCL
- QOCS
- QSimulator.jl
- QTop
- QWIRE
- QImp
- Qrack
- QuEST
- QuSim
- QuTiP
- QuaC
- Quantum Circuit

Quantum full-stack libraries

- Cirq
- Forest
- Ocean
- ProjectQ
- Q#
- Qiskit
- Qiskit-JS
- Strawberry Fields
- XACC

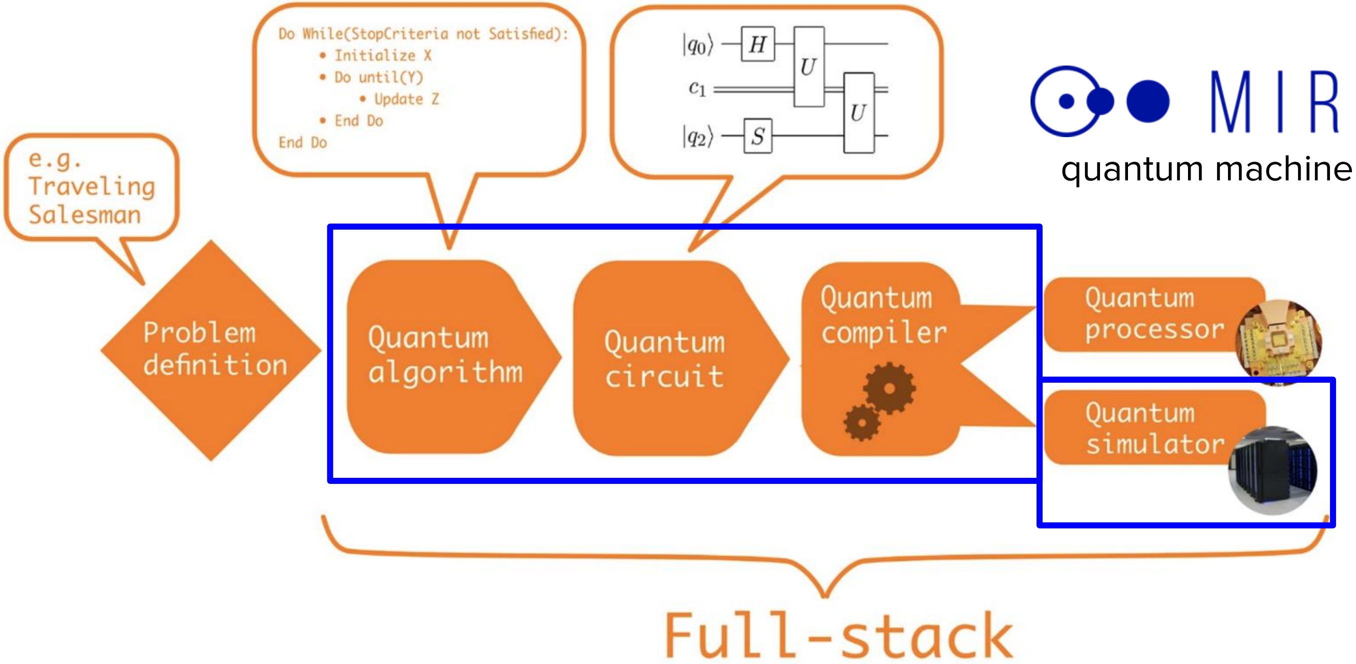
Quantum algorithms

- Adapt
- FermiLib
- Grove
- OpenFermion
- PennyLane
- QFog
- Qiskit Aqua
- Qiskit Tutorial
- Quantum Katas
- QuantumFlow
- QuantumTomography.jl
- Quantum_Edward
- XACC Examples
- XACC QChem
- XACC VQE

Quantum fun

- Entanglion
- QSEL
- Quantum Awesomeness
- Quantum Battleships
- Quantum Catsweeper
- Quantum Music Composer for IBM Q
- Quantum Music Composer for Rigetti

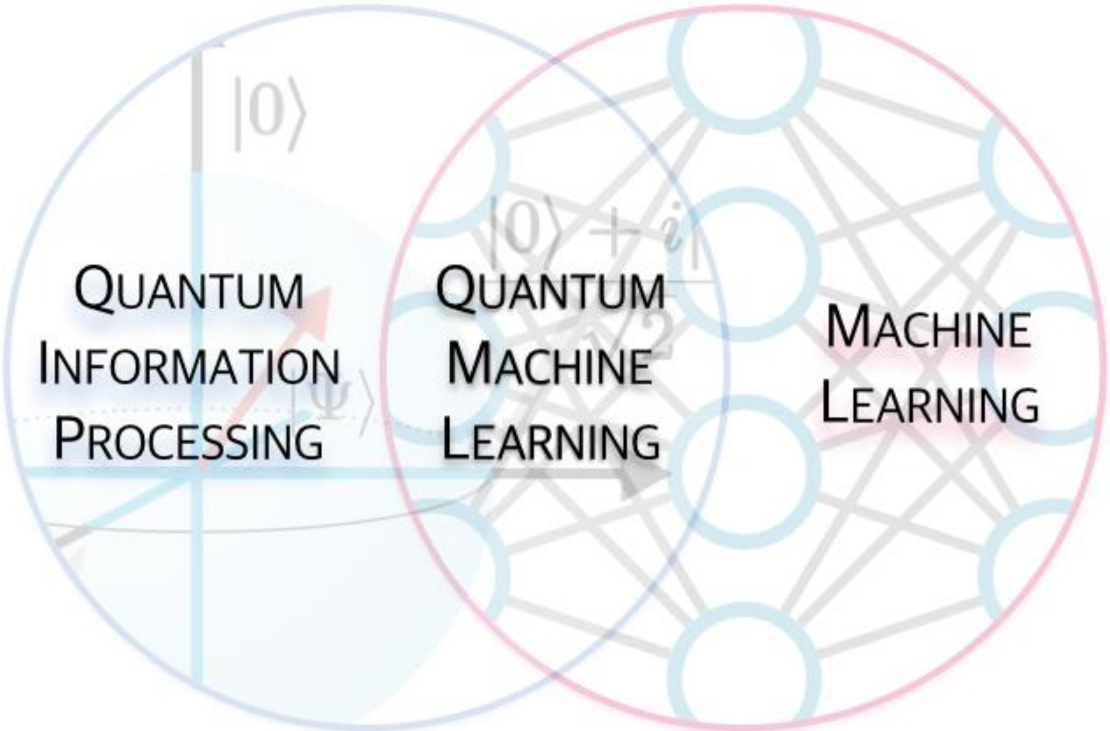
Software

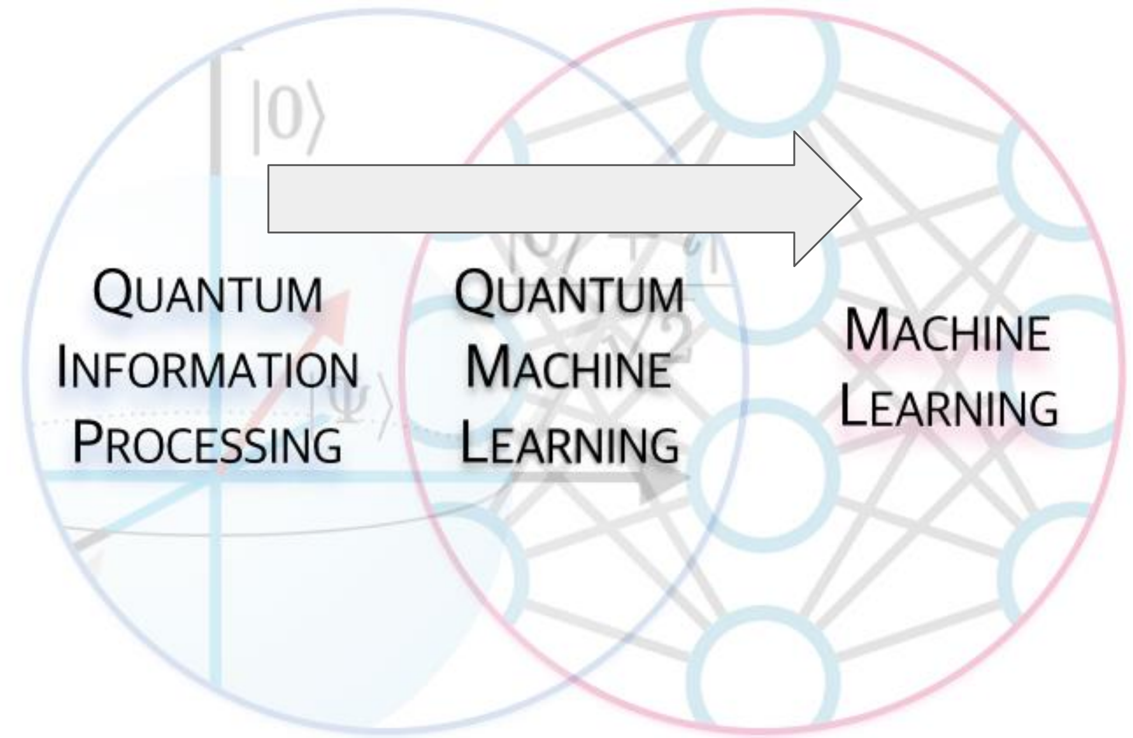


MIRAEX

quantum machine learning

M. Fingerhuth, T. Babej, P. Wittek. (2018). Open source software in quantum computing. *PLOS ONE* 13(12):e0208561.


QUANTUM INFORMATION PROCESSING
QUANTUM MACHINE LEARNING
MACHINE LEARNING


QUANTUM INFORMATION PROCESSING
QUANTUM MACHINE LEARNING
MACHINE LEARNING

Quantum Machine Learning landscape

GATE MODEL QC

ADIABATIC

Quantum Machine Learning landscape

GATE MODEL QC

ADIABATIC

Quantum Annealing

map to Ising model

CQ: classical data, quantum algo

optimization problems, qBoost, clustering, sampling a thermal state

Quantum Machine Learning landscape

GATE MODEL QC

ADIABATIC

Quantum Annealing

map to Ising model

CQ: classical data, quantum algo

optimization problems, qBoost, clustering, sampling a thermal state

Quantum Machine Learning landscape

GATE MODEL QC

qBLAS:
quantum basic linear algebra subroutines

algorithm focus: can we do linear algebra faster, on perfect hardware?
QQ: quantum data, quantum algo

Mostly HHL-based, QFT, QPE, Quantum Matrix Inversion, qPCA, qSVM

ADIABATIC

Quantum Annealing

map to Ising model
CQ: classical data, quantum algo

optimization problems, qBoost, clustering, sampling a thermal state

Quantum Machine Learning Algorithms: Read the Fine Print

Scott Aaronson

Solve Ax = b in logarithmic time (b = length n):

- load data fast enough / quantum ram
- unitary operations exp(-iAt) efficient
- A is ‘well-conditioned’
- don’t need full access to x

if one condition not met ➔ speedup gone!

‘big data’, ‘quantum speedup’, ‘revolutionize computing’…

Quantum Machine Learning landscape

GATE MODEL QC

qBLAS:
quantum basic linear algebra subroutines

algorithm focus: can we do linear algebra faster, on perfect hardware?
QQ: quantum data, quantum algo

Mostly HHL-based, QFT, QPE, Quantum Matrix Inversion, qPCA, qSVM

NISQ:
noisy intermediate scale quantum devices

hardware focus: can we do it different, with current (noisy) hardware?
CQ: classical data, quantum algo

variational circuits, QAOA, kernel methods, ‘qNN’, approximate thermalization

ADIABATIC

Quantum Annealing

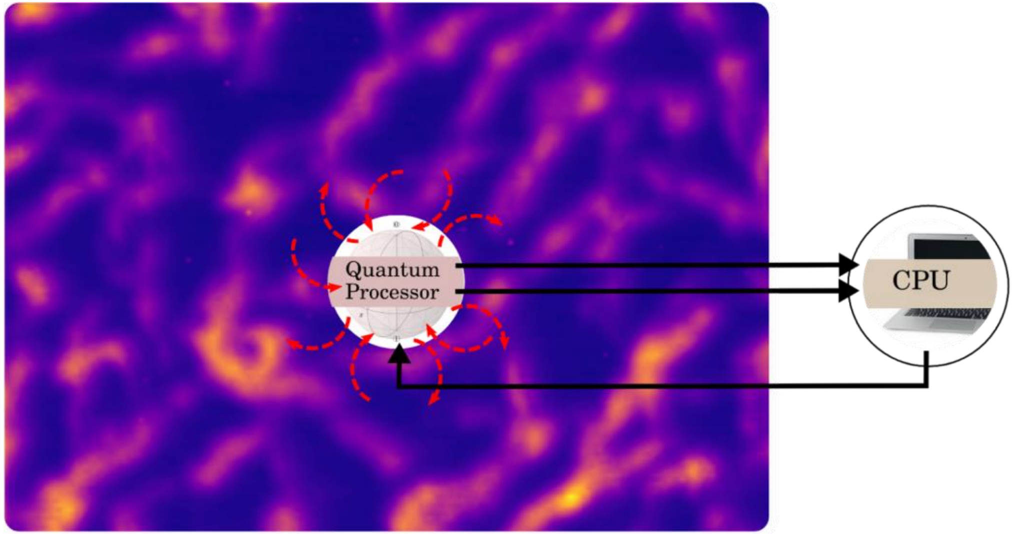
map to Ising model
CQ: classical data, quantum algo

optimization problems, qBoost, clustering, sampling a thermal state

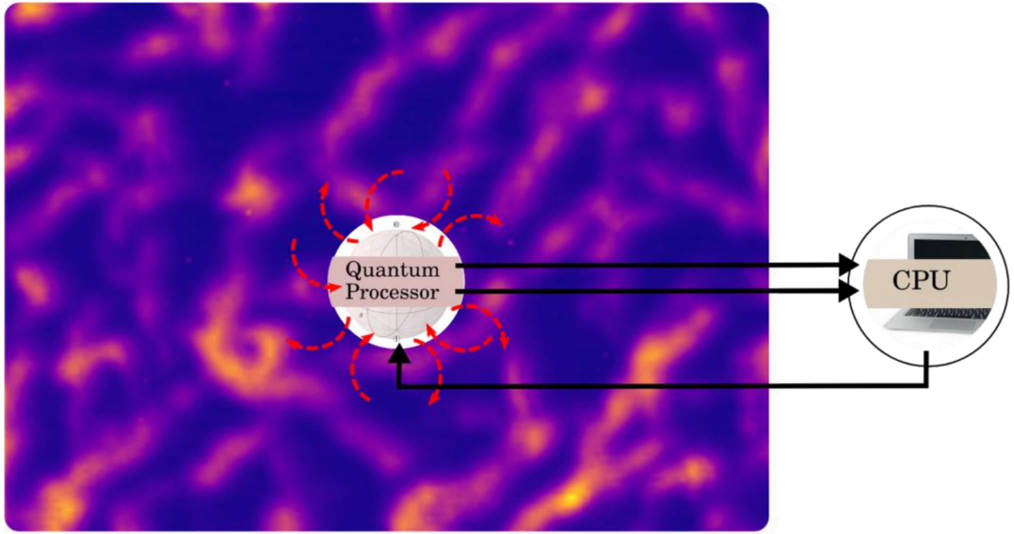
Quantum Machine Learning landscape

GATE MODEL QC	**qBLAS:** *quantum basic linear algebra subroutines*	**algorithm focus**: can we do linear algebra faster, on perfect hardware? QQ: quantum data, quantum algo *Mostly HHL-based, QFT, QPE, Quantum Matrix Inversion, qPCA, qSVM*
	NISQ: *noisy intermediate scale quantum devices*	**hardware focus**: can we do it different, with current (noisy) hardware? CQ: classical data, quantum algo *variational circuits, QAOA, kernel methods, 'qNN', approximate thermalization*
ADIABATIC	**Quantum Annealing**	map to Ising model CQ: classical data, quantum algo *optimization problems, qBoost, clustering, sampling a thermal state*

Exploiting NISQ hardware: Variational Circuits



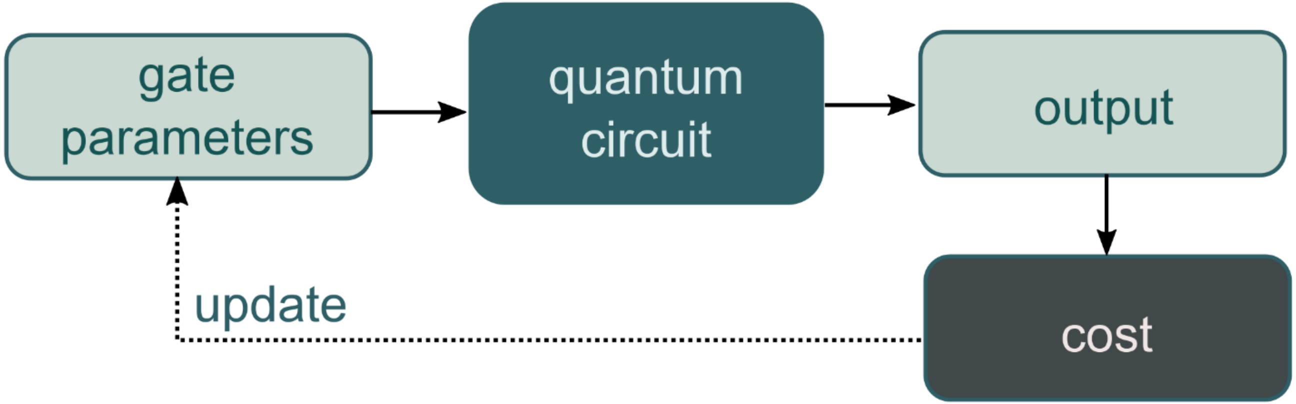
Exploiting NISQ hardware: Variational Circuits



Exploiting NISQ hardware: Variational Circuits

run a short parametrized (shallow) circuit on QPU & optimize on CPU

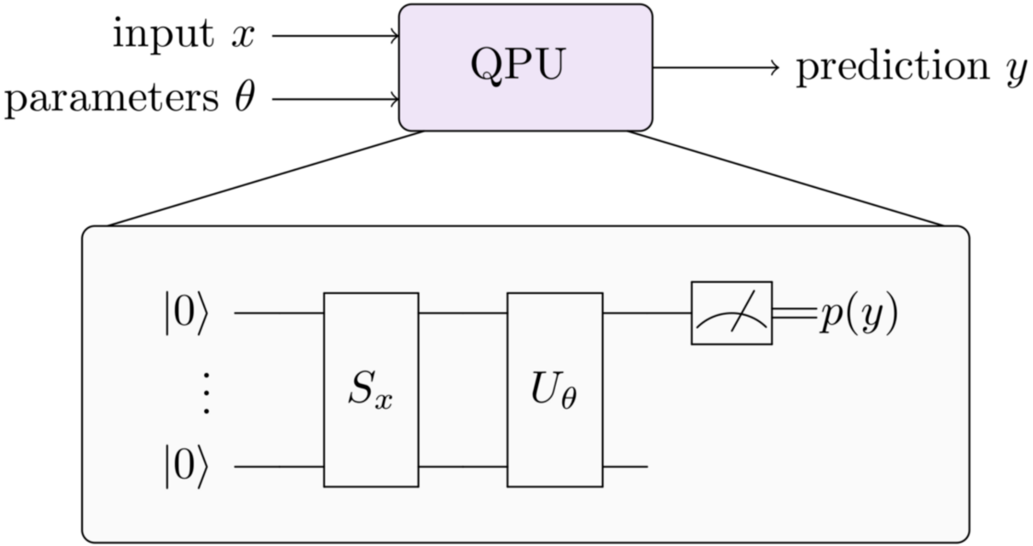
designed for noisy & imperfect quantum computers



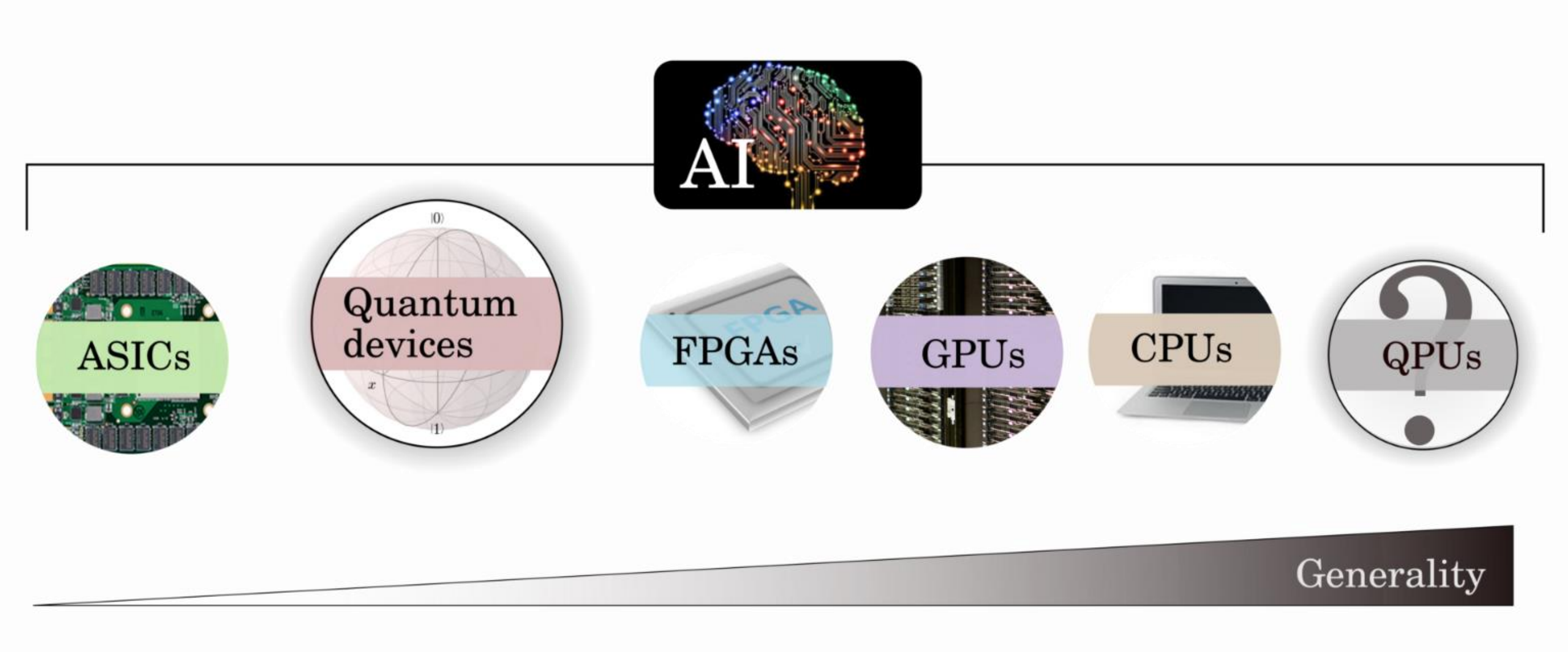
quantum neural network

"A quantum neural network is any quantum circuit with trainable continuous parameters".

Variational Circuits - QPU



Variational Circuits - QPU



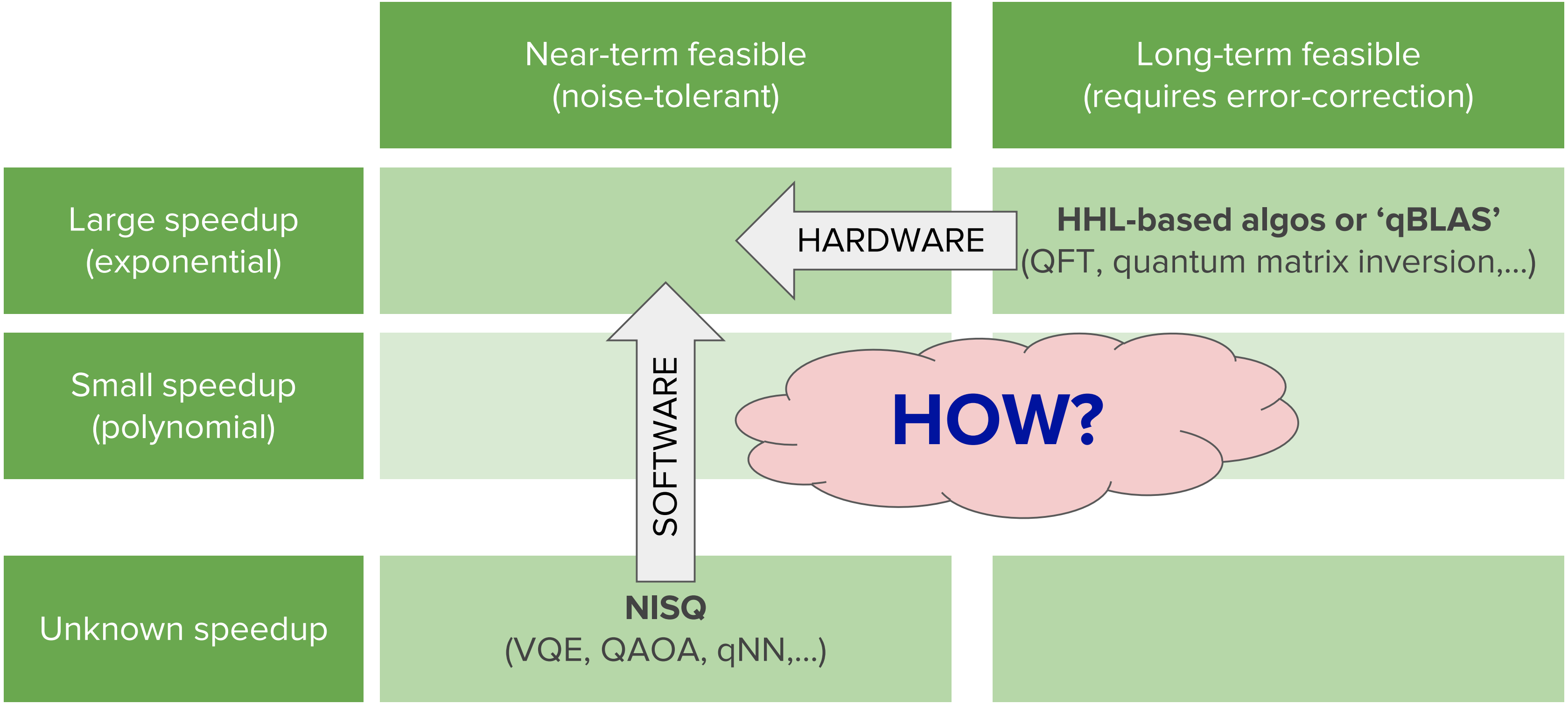
Quantum Machine Learning landscape

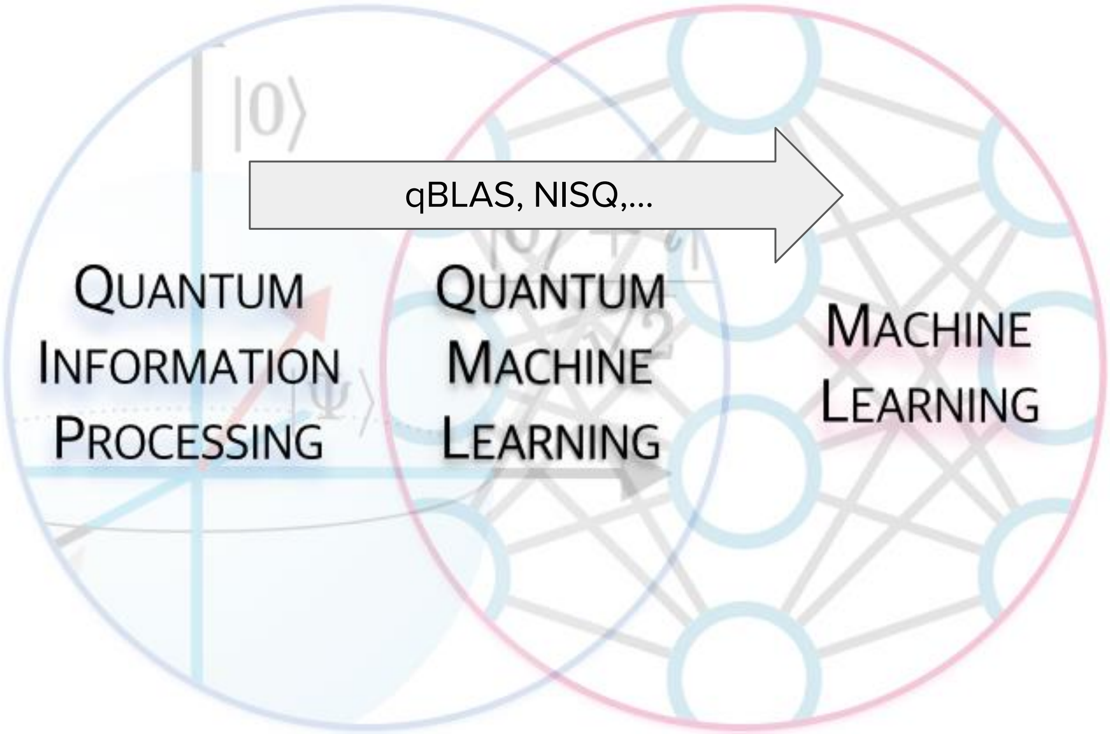
	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)		**HHL-based algos or 'qBLAS'** (QFT, quantum matrix inversion,...)
Small speedup (polynomial)		
Unknown speedup	**NISQ** (VQE, QAOA, qNN,...)	

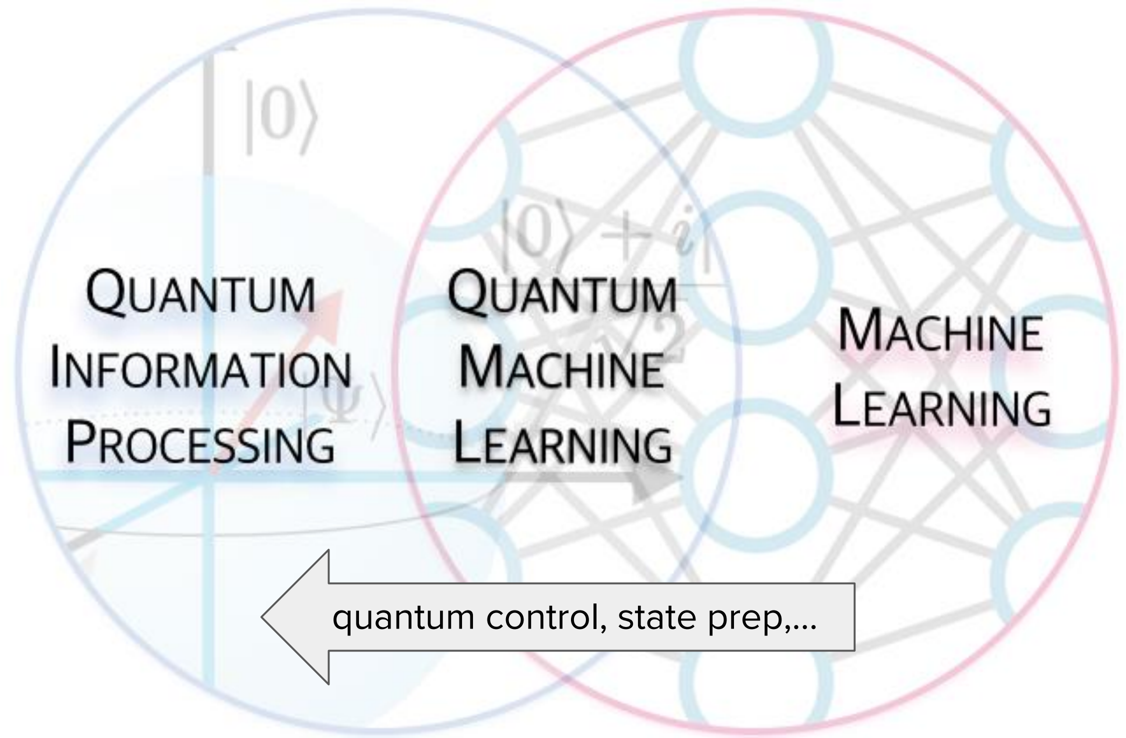
Quantum Machine Learning landscape

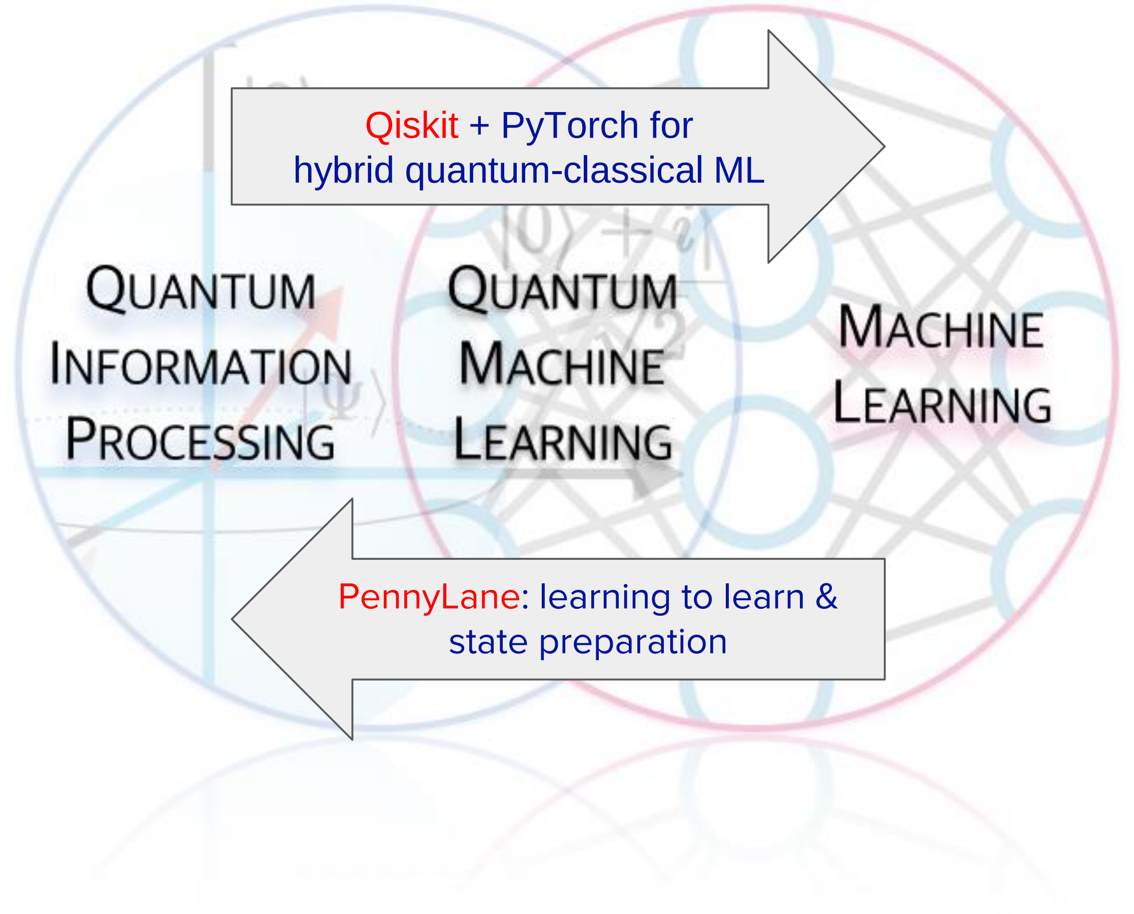
	Near-term feasible (noise-tolerant)	Long-term feasible (requires error-correction)
Large speedup (exponential)		**HHL-based algos or ‘qBLAS’** (QFT, quantum matrix inversion,...)
Small speedup (polynomial)		
Unknown speedup	**NISQ** (VQE, QAOA, qNN,...)	

Quantum Machine Learning landscape

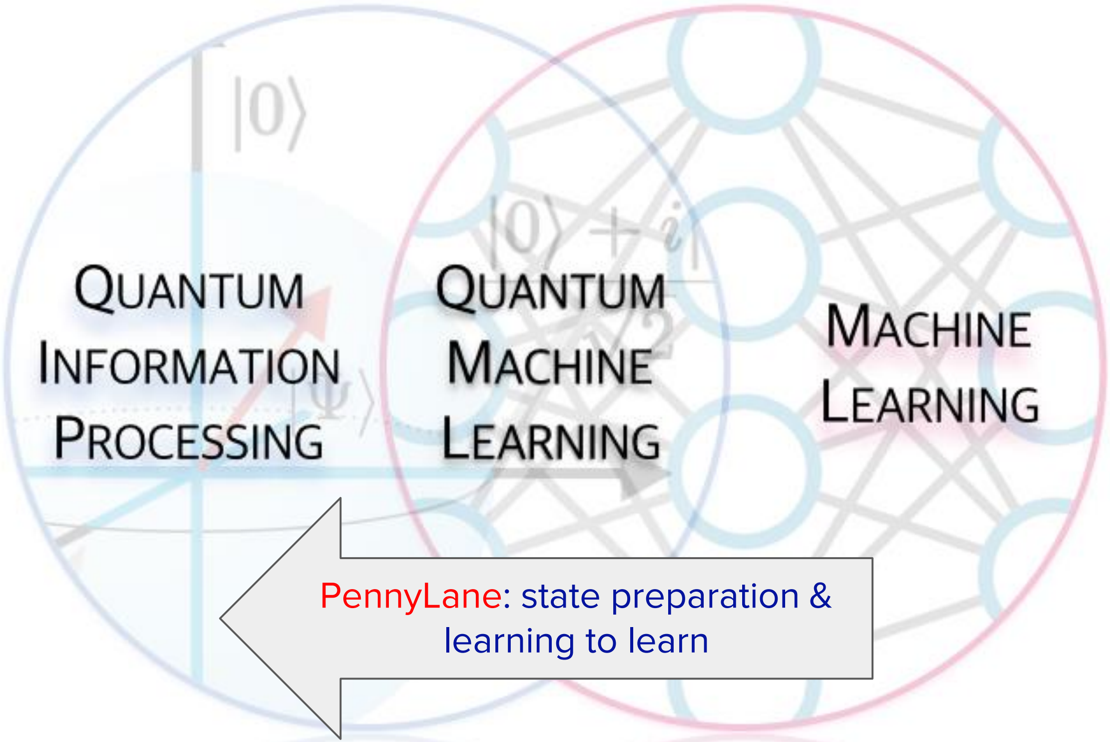



qBLAS, NISQ,...
QUANTUM INFORMATION PROCESSING
QUANTUM MACHINE LEARNING
MACHINE LEARNING


QUANTUM INFORMATION PROCESSING
QUANTUM MACHINE LEARNING
MACHINE LEARNING
quantum control, state prep,...


Qiskit + PyTorch for
hybrid quantum-classical ML
QUANTUM
INFORMATION
PROCESSING
QUANTUM
MACHINE
LEARNING
MACHINE
LEARNING
PennyLane: learning to learn &
state preparation

MIRAEX



PennyLane: state preparation & learning to learn

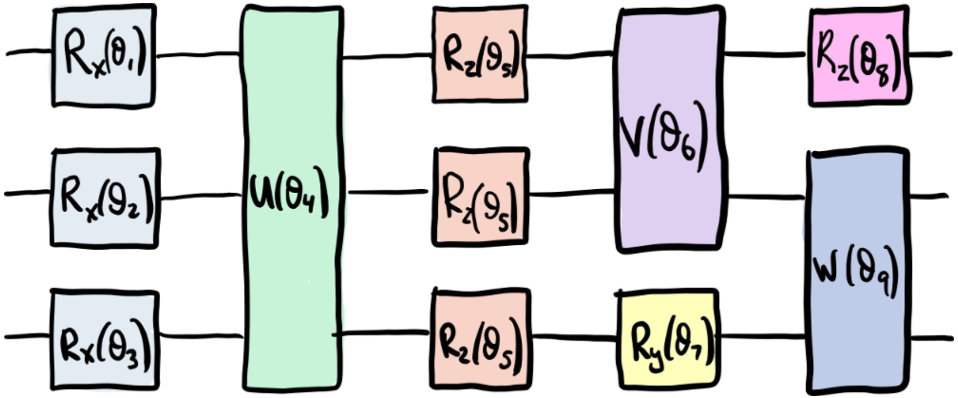
Meta-learning few-shot optimization for Quantum Circuits

CREATIVE DESTRUCTION LAB

Blog Post

Founders solve problems from leading quantum computing hardware and software companies

MIRAEX

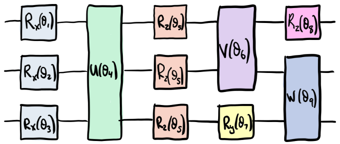


Quantum Neural Networks (QNNs)
are promising,

but still facing challenges.

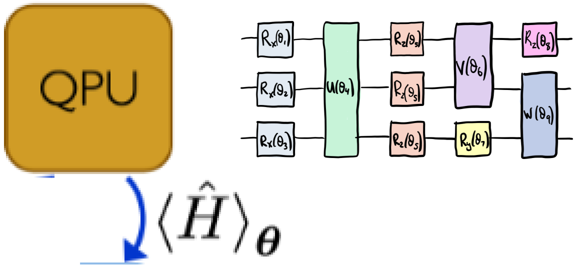
Hybrid variational quantum algorithms

- Parametric quantum circuit
- Change parameter, measure observable
- Rinse repeat to optimize the observable



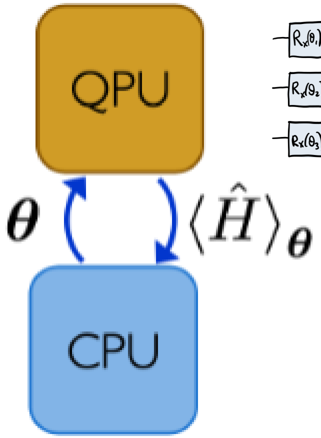
Hybrid variational quantum algorithms

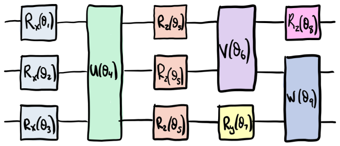
- Parametric quantum circuit
- Change parameter, measure observable
- Rinse repeat to optimize the observable



Hybrid variational quantum algorithms

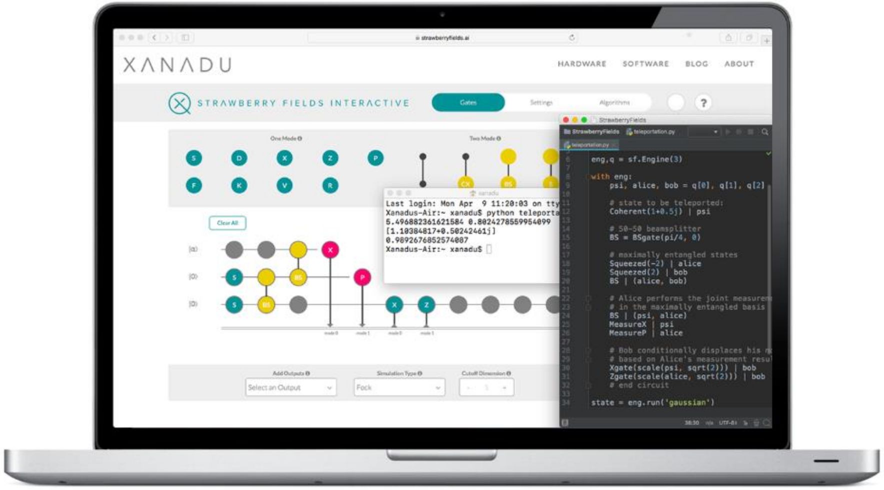
- Parametric quantum circuit
- Change parameter, measure observable
- Rinse repeat to optimize the observable





STRAWBERRY FIELDS

Open-source software for photonic quantum computing



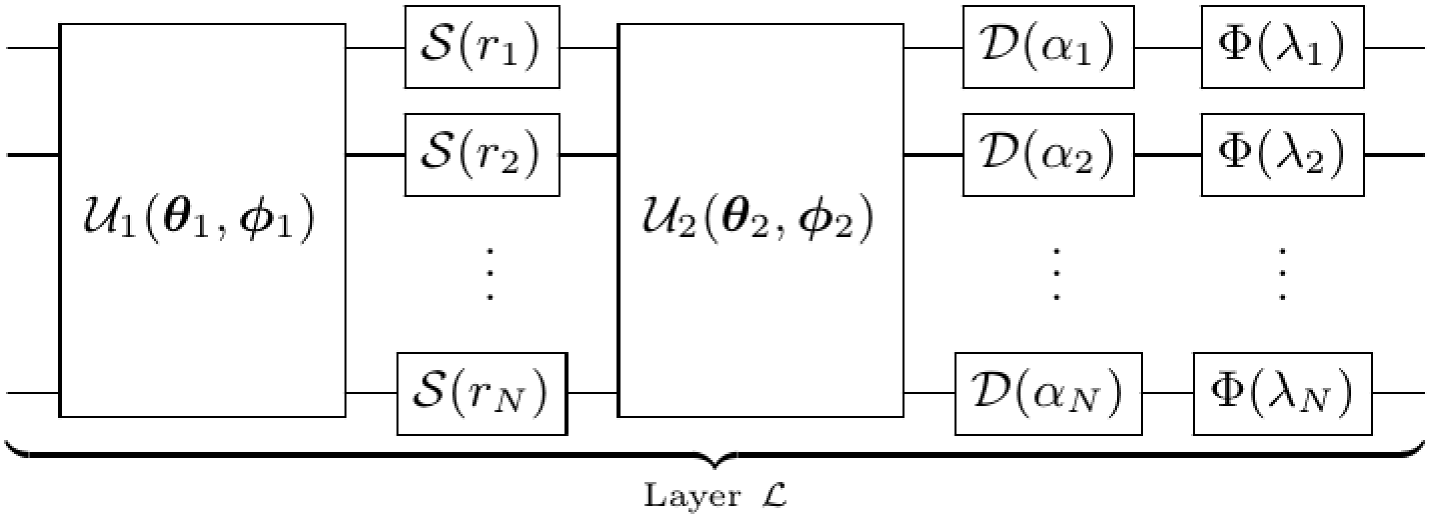
TensorFlow

to support backpropagation through quantum simulations on GPUs

=

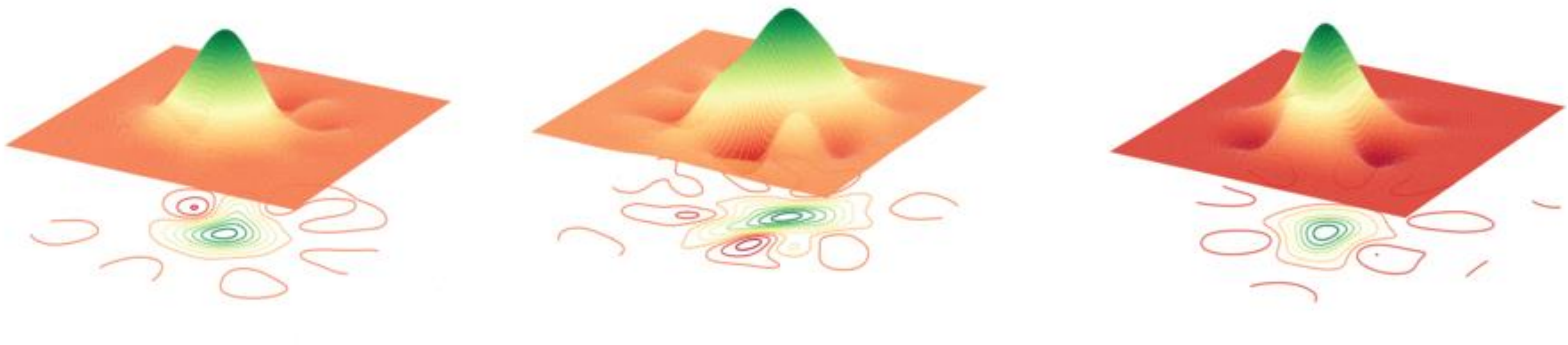
deep learning to design and optimize circuits!

Use variational circuits to learn state preparation & gate implementation



Machine learning method for state preparation and gate synthesis on photonic quantum computers (https://arxiv.org/abs/1807.10781)

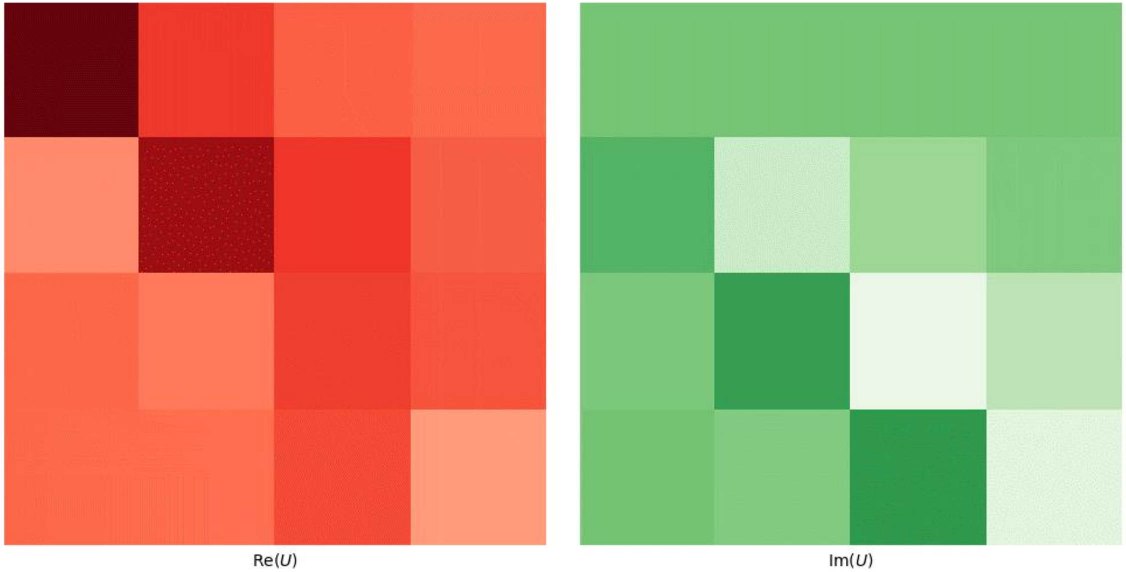
Use variational circuits to learn state preparation & gate implementation



single photon state | Schrödinger Cat state | ON state

Machine learning method for state preparation and gate synthesis on photonic quantum computers (https://arxiv.org/abs/1807.10781**)**

Use variational circuits to learn state preparation & gate implementation



Machine learning method for state preparation and gate synthesis on photonic quantum computers (https://arxiv.org/abs/1807.10781)

Hybrid variational quantum algorithms

- Parametric quantum circuit
- Change parameter, measure observable
- Rinse repeat to optimize the observable

Need:

- **Initial parameter set**
- **Local optimization: how to minimize?**

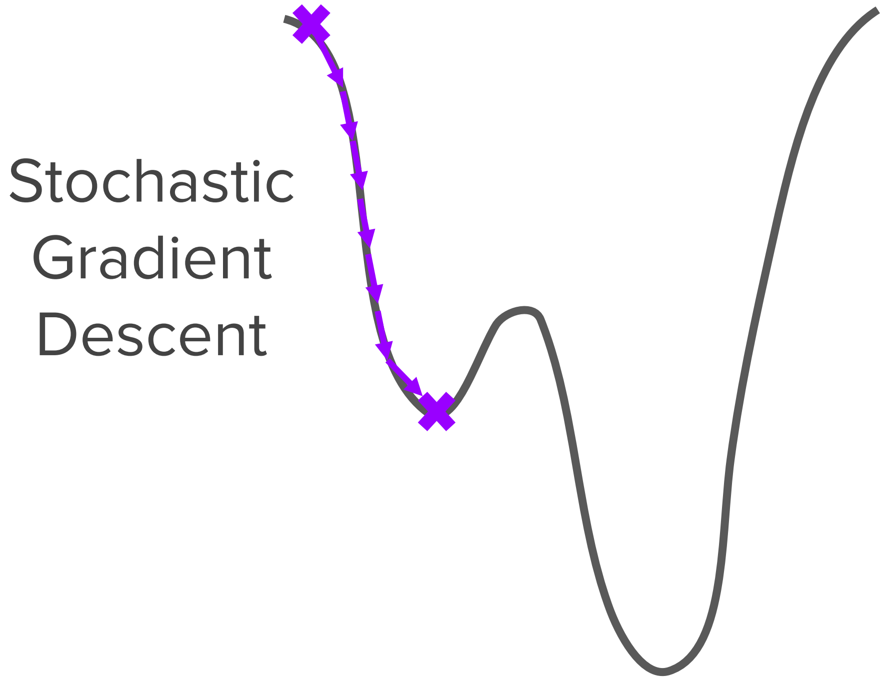
PENNYLANE PyTorch

Our challenge: finding good parameter initialization heuristics that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.

MIRAEX

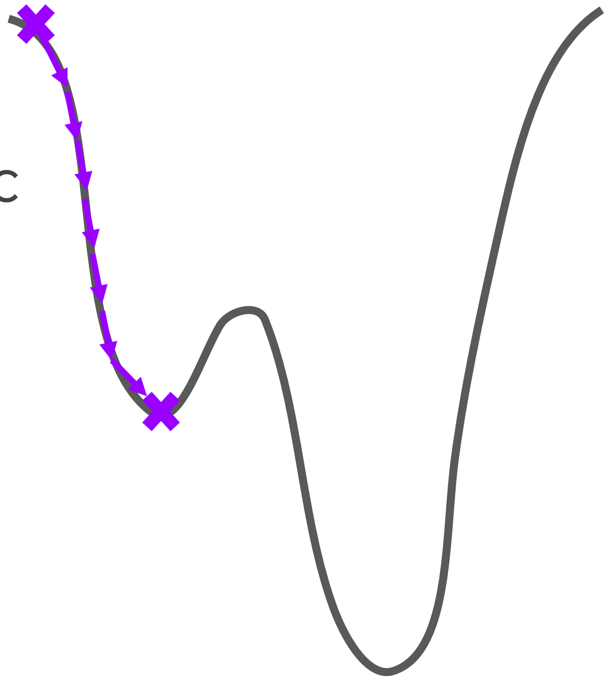
@kareldumon

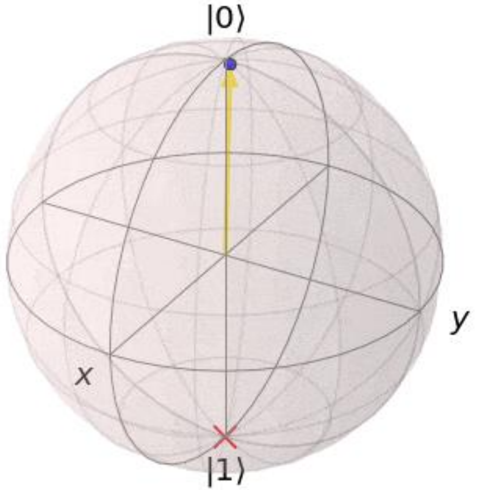
Our challenge: finding good parameter initialization heuristics that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.



Our challenge: finding good parameter initialization heuristics that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.

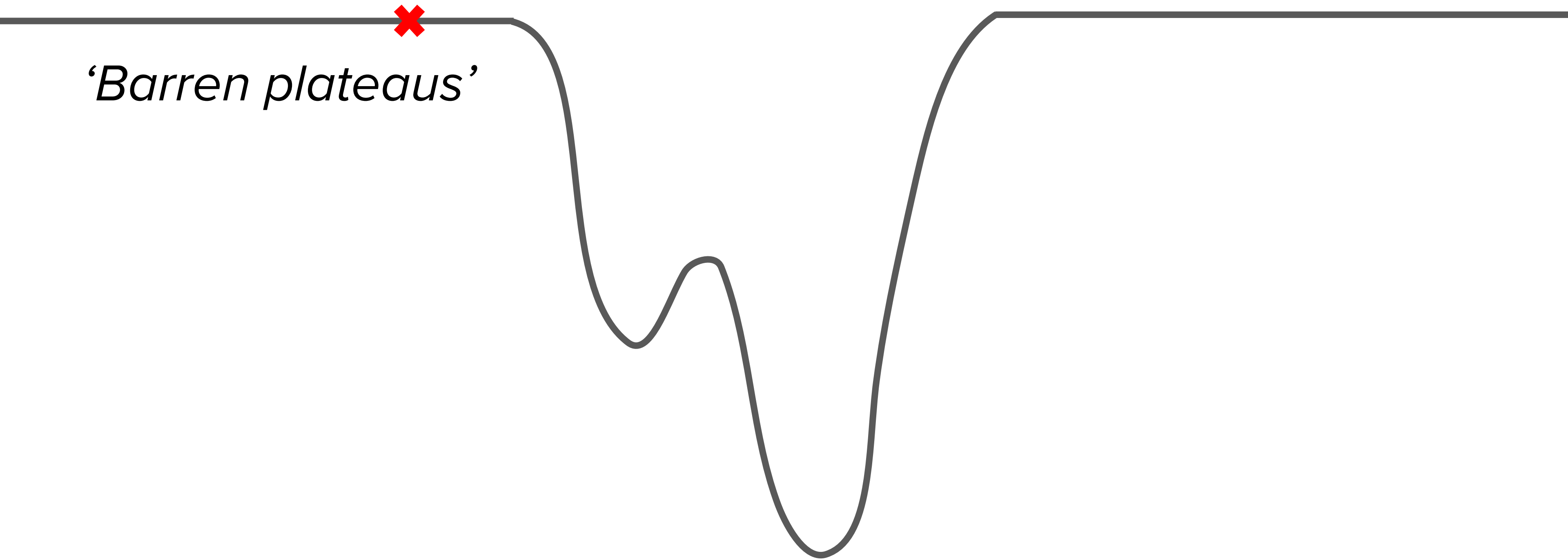
Stochastic Gradient Descent



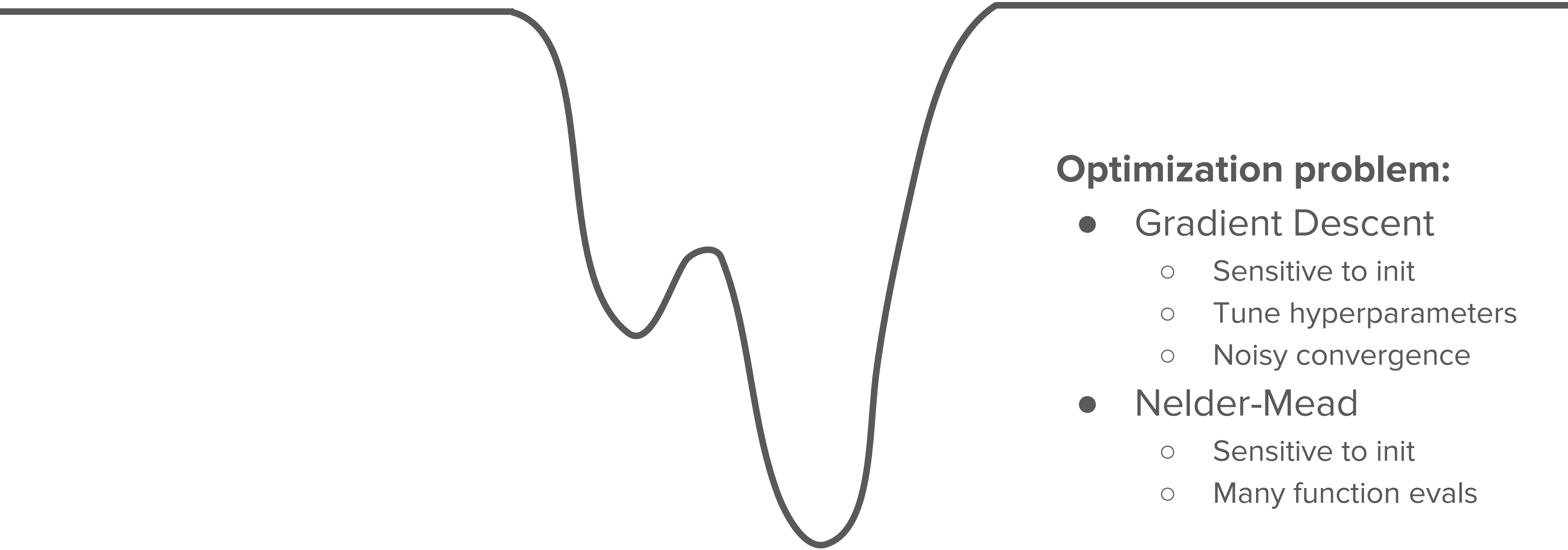


@kareldumon

Our challenge: finding good parameter initialization heuristics that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.



Our challenge: finding good parameter initialization heuristics that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.



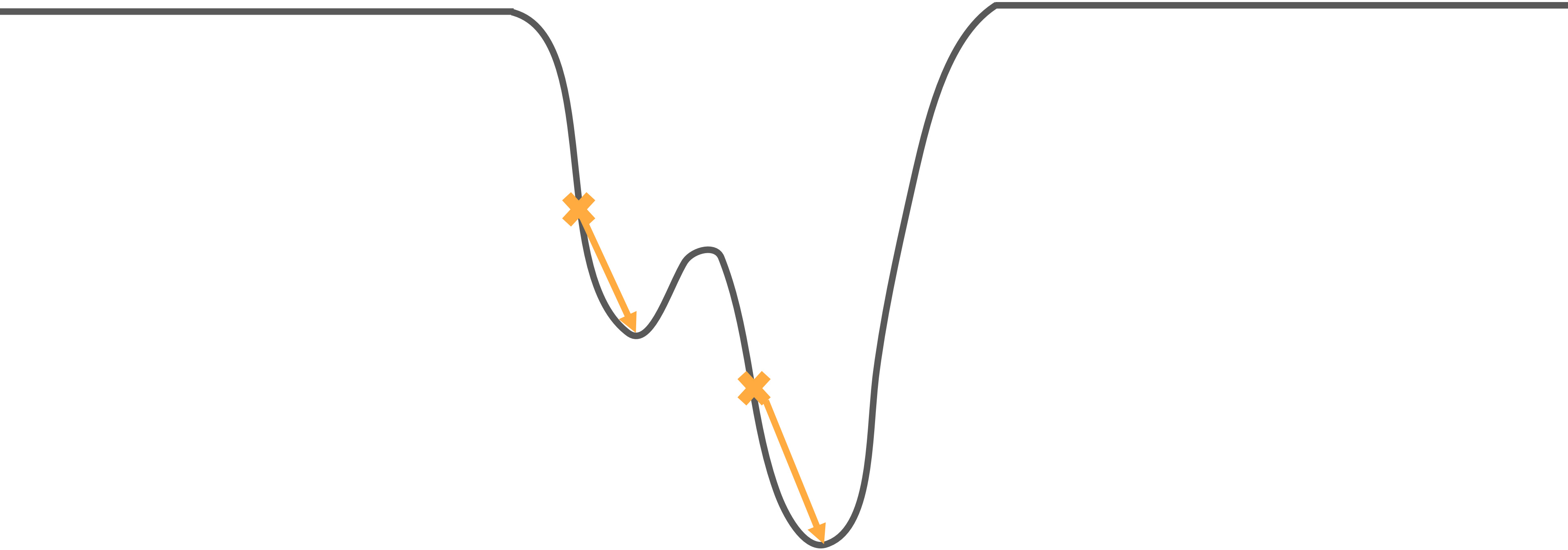
Optimization problem:

- Gradient Descent
  - Sensitive to init
  - Tune hyperparameters
  - Noisy convergence
- Nelder-Mead
  - Sensitive to init
  - Many function evals

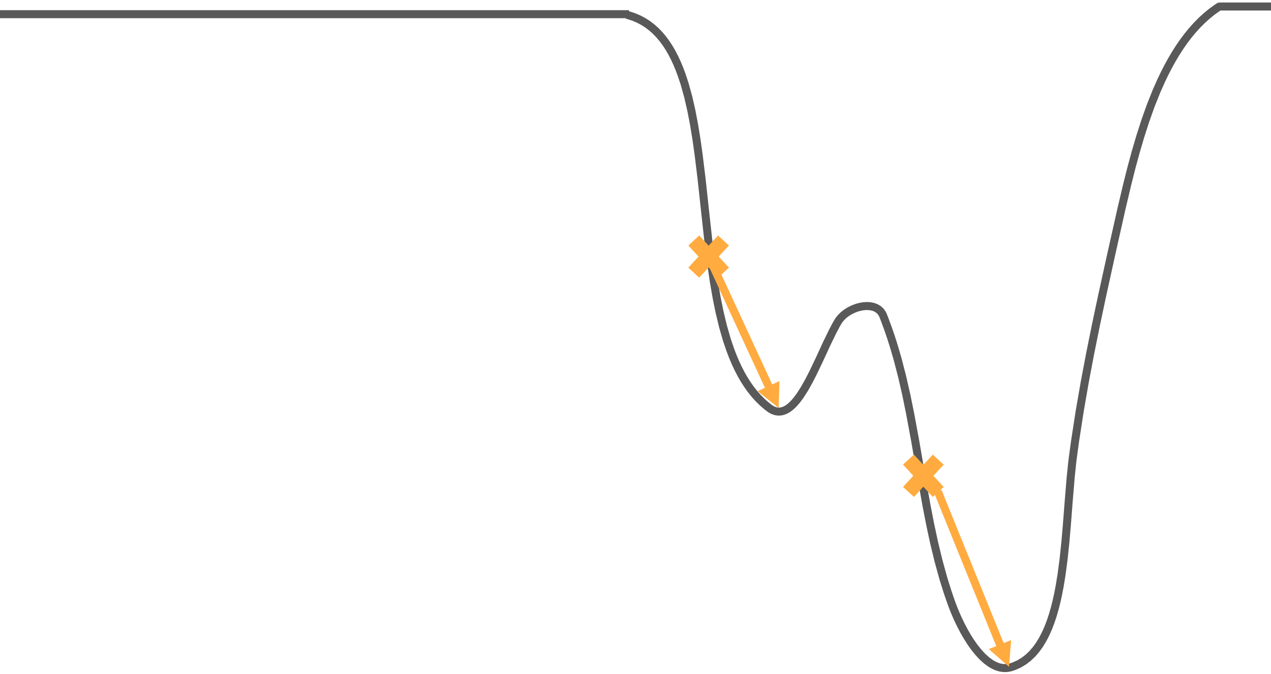
MIRAEX

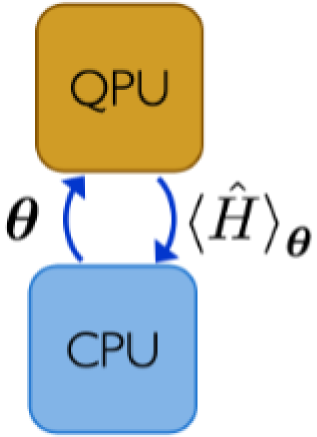
@kareldumon

Our challenge: finding good **parameter initialization heuristics** that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.



Our challenge: finding good **parameter initialization heuristics** that ensure rapid and consistent convergence to local minima of the parameterized quantum circuit landscape.

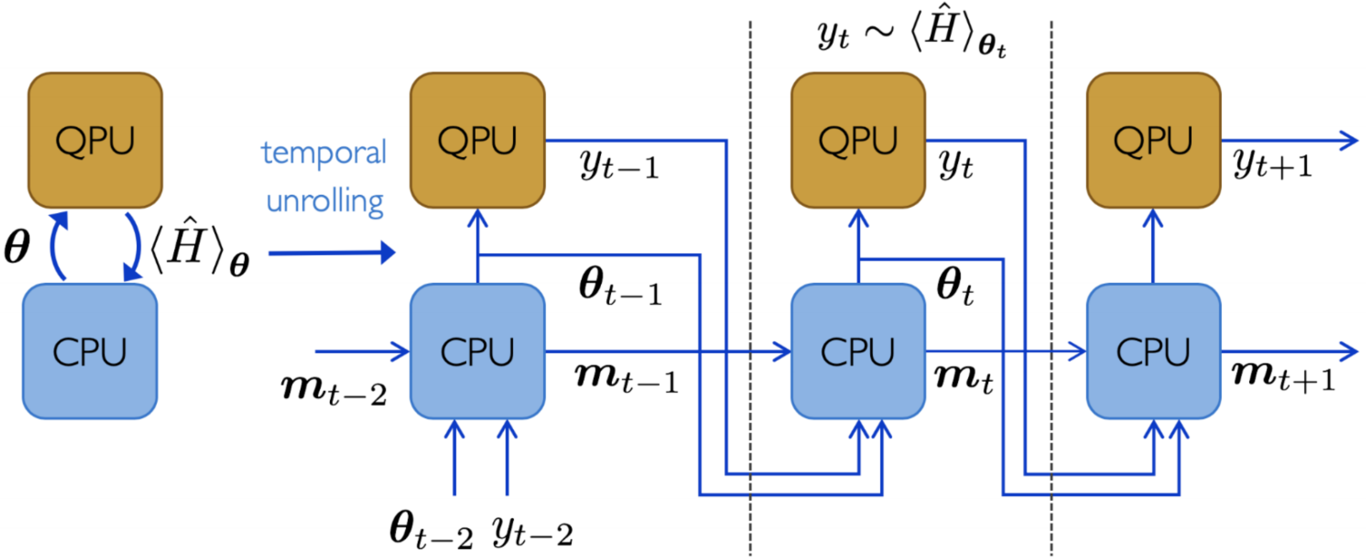




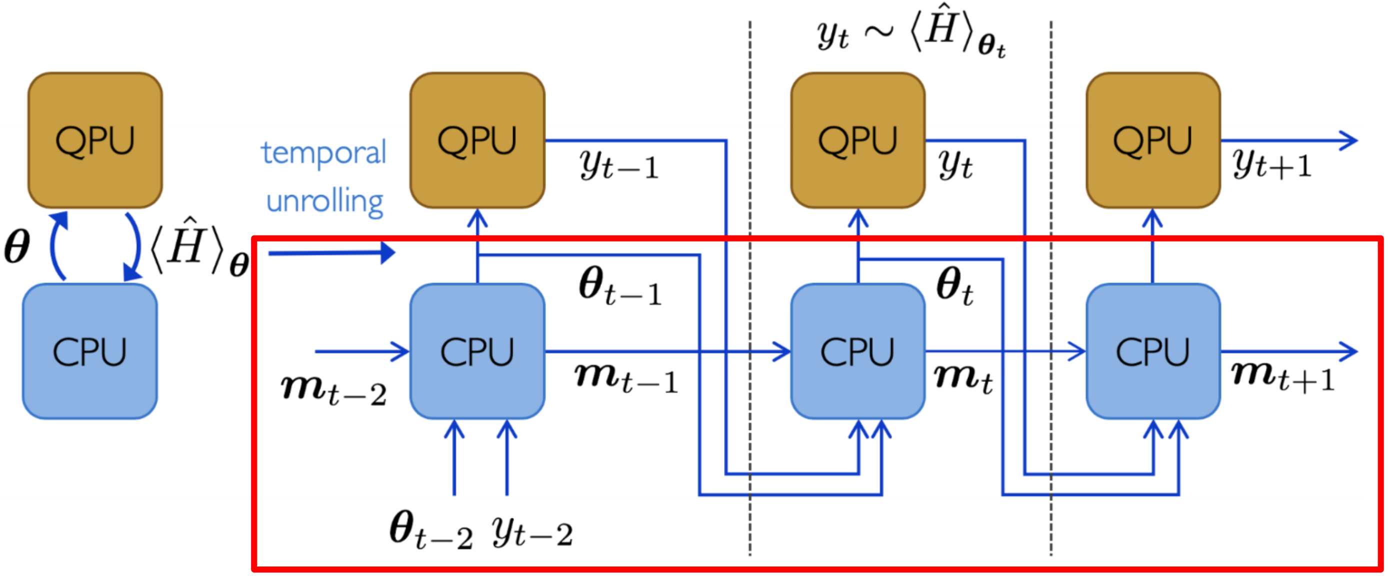
+ make it less costly & scalable?

@kareldumon

Current status: general hybrid variational quantum algorithms

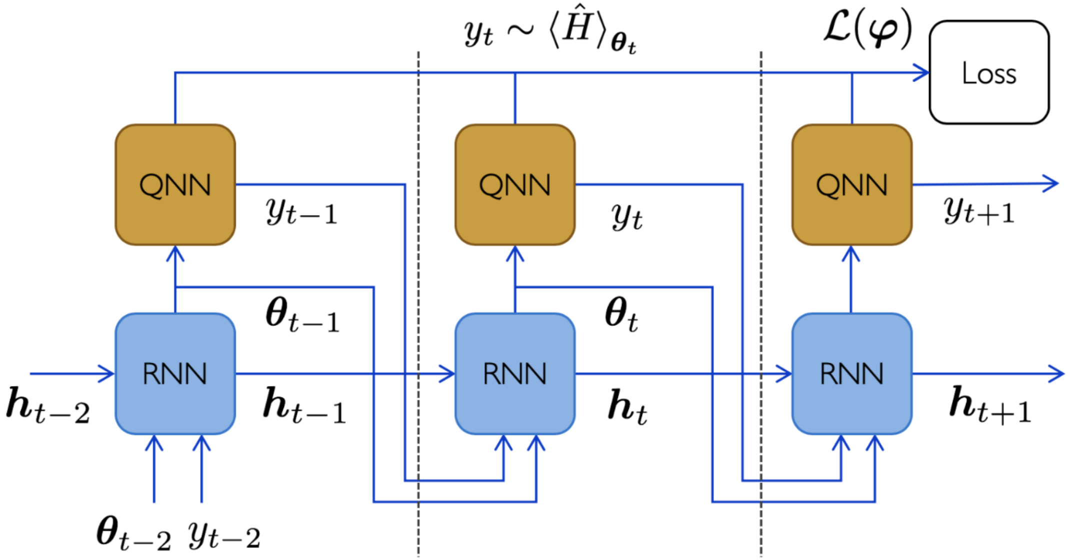


Current status: general hybrid variational quantum algorithms



looks familiar!

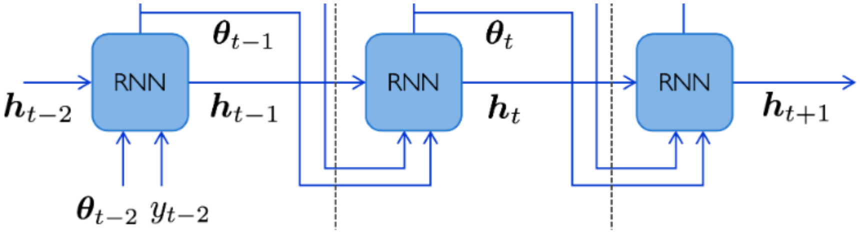
Our solution: train **classical neural networks** to assist in the quantum learning process to rapidly find approximate optima in the parameter landscape, i.e. *meta-learning.*



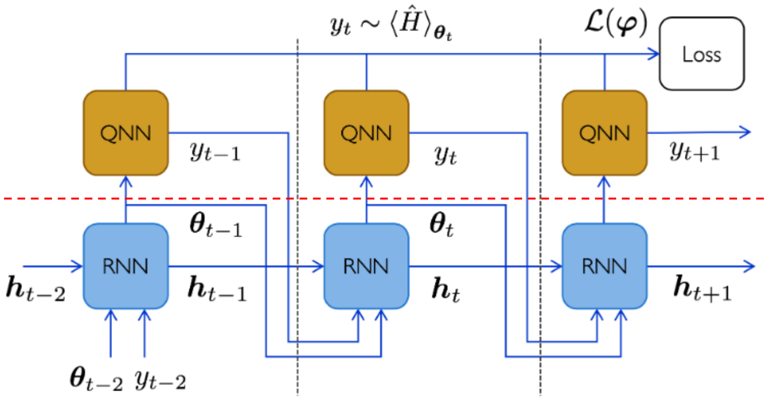
Learning to learn with quantum neural networks via classical neural networks, Guillaume Verdon et. al., (2019) [*arXiv:1907.05415*]

About the RNN:

- Train your RNN like a (QU)BOSS
  - Learns init naturally
  - Learns to find good neighborhood quickly, but needs further local optimization
  - One LSTM layer
  - Choose a loss that incentivizes beneficial updates



Our architecture:



rigetti IBM Q

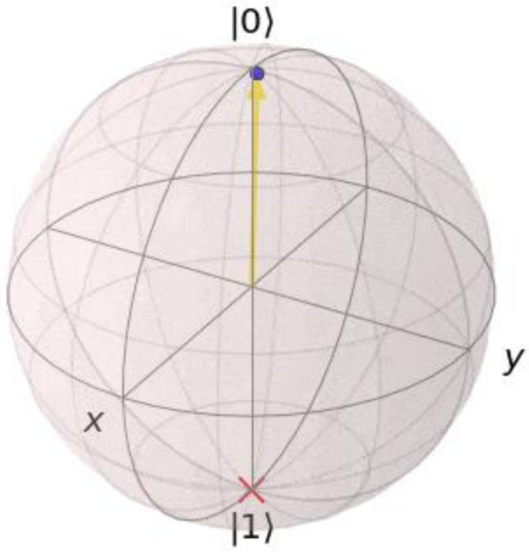
simulator / QPU

PENNYLANE

PyTorch

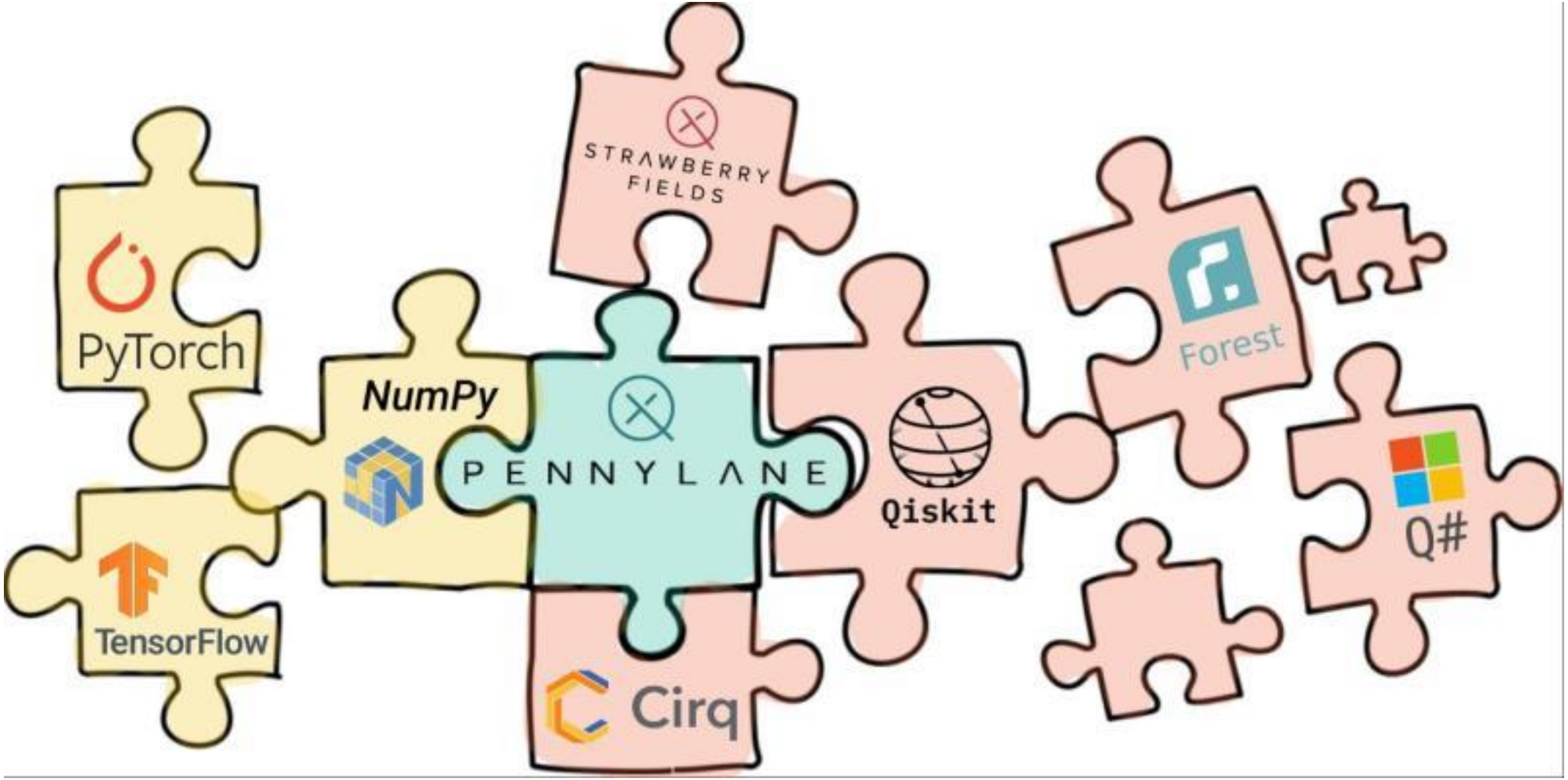
XANADU

```python
import pennylane as qml
import torch
from torch.autograd import Variable

qpu = qml.device('forest.qpu', device='Aspen-1-2Q-B')

@qml.qnode(dev, interface='torch')
def circuit(phi, theta):
    # Quantum node running on the Rigetti QPU
    qml.RX(theta, wires=0)
    qml.RZ(phi, wires=0)
    return qml.expval.PauliZ(0)

def cost(phi, theta, step):
    # Classical node
    target = -(-1)**(step // 100)
    return torch.abs(circuit(phi, theta) - target)**2

phi = Variable(torch.tensor(1.), requires_grad=True)
theta = Variable(torch.tensor(0.05), requires_grad=True)
opt = torch.optim.Adam([phi, theta], lr = 0.1)

for i in range(400):
    opt.zero_grad()
    loss = cost(phi, theta, i)
    loss.backward()
    opt.step()
```



PENNYLANE PyTorch TensorFlow

STRAWBERRY FIELDS rigetti Forest Qiskit


STRAWBERRY FIELDS
PyTorch
NumPy
PENNYLANE
Qiskit
Forest
Q#
TensorFlow
Cirq

Our training details:

Trained the RNN optimizer

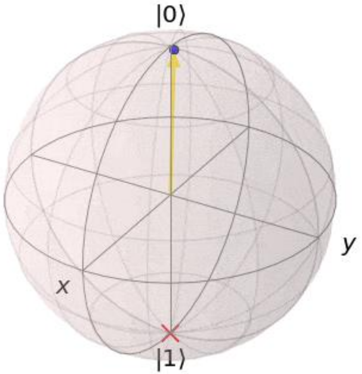
- 2 qubit rotation, VQE, QAOA
- Via quantum simulation in (noiseless) and PyTorch

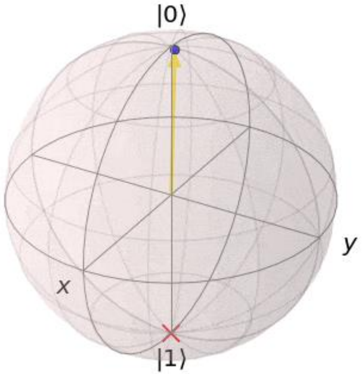
Tested

- Quantum simulation
- Rigetti HW
  PennyLane → ‘one flip of the switch’!
- IBM HW

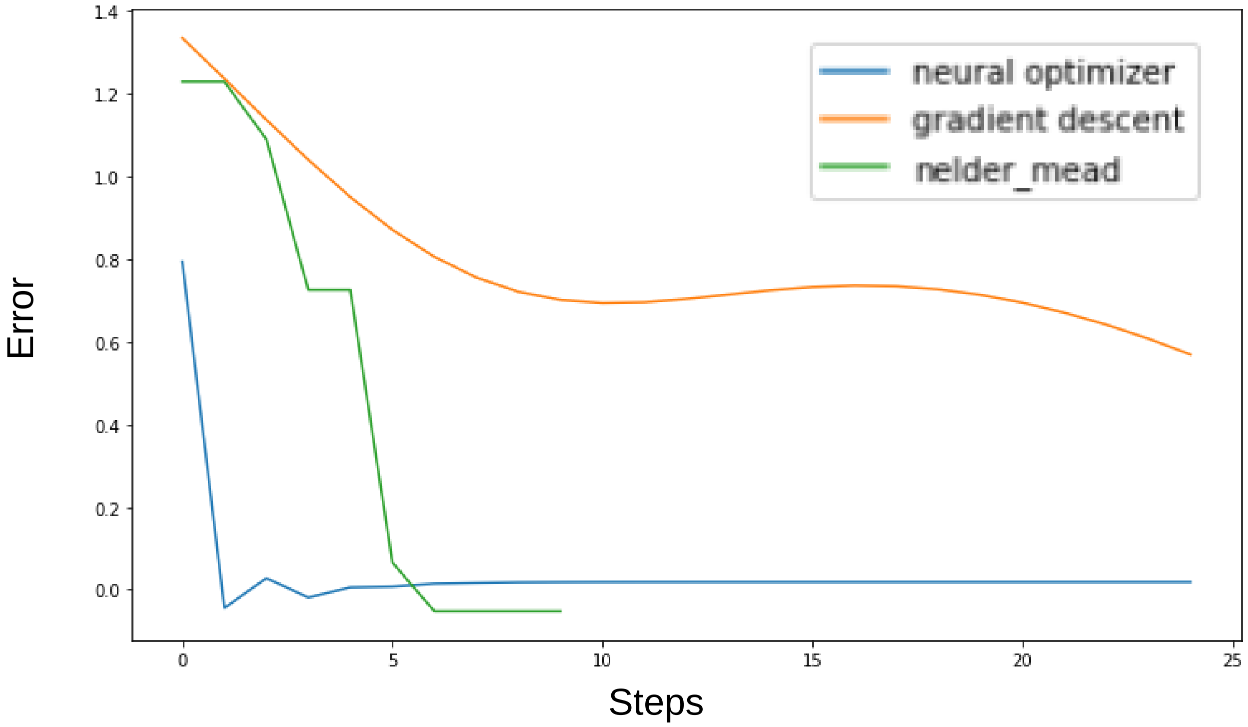
Example 1: 2 qubit rotation / state preparation

- Simulated
- Limit the func evals in NM to the same number
- Different init for GD, NM, different targets





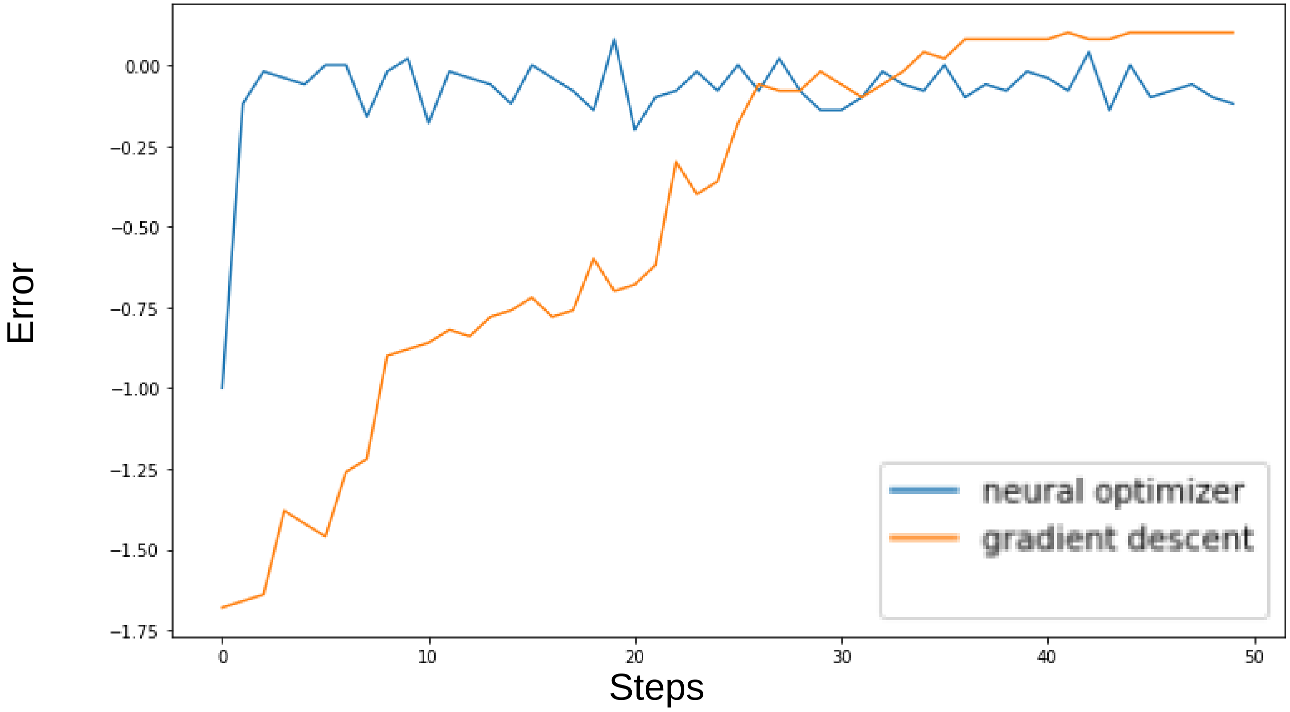
Example 1: 2 qubit rotation



MIRAEX

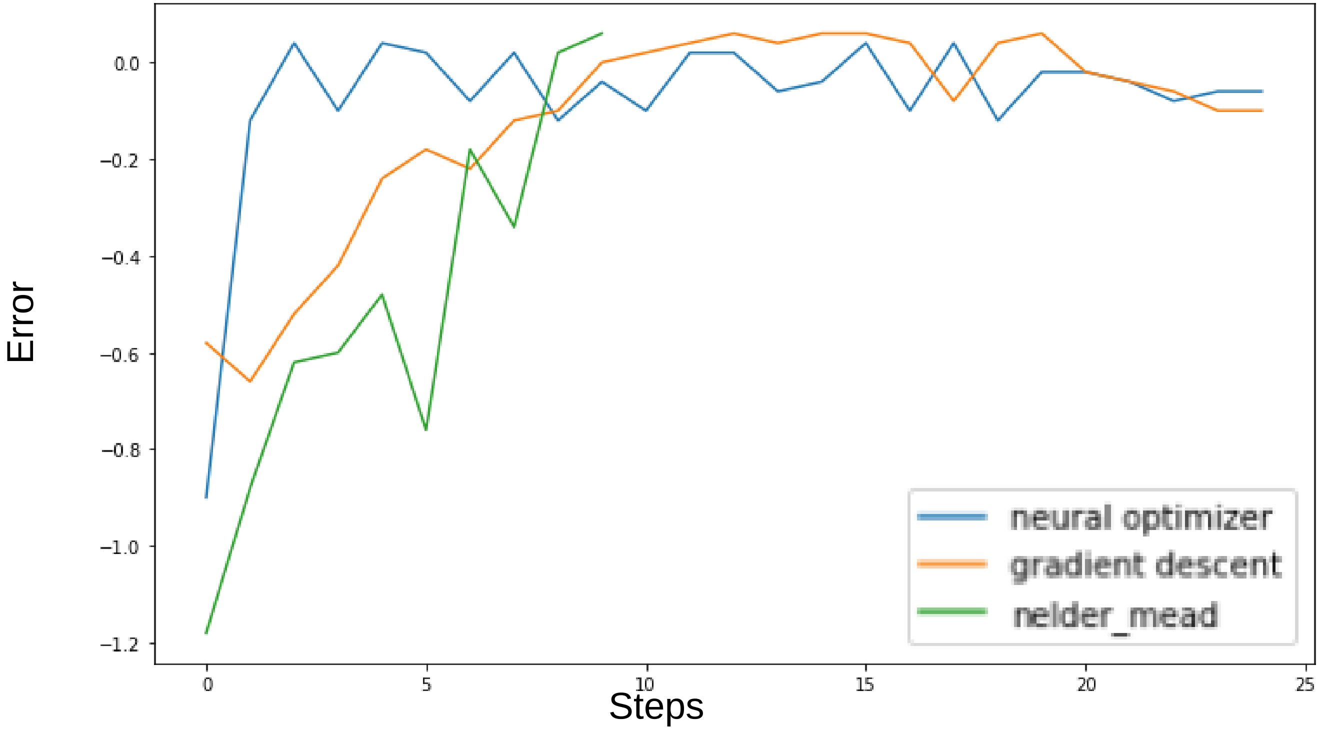
@kareldumon

Example 1: 2 qubit rotation



@kareldumon

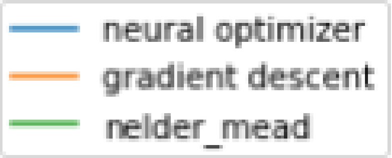
Example 1: 2 qubit rotation

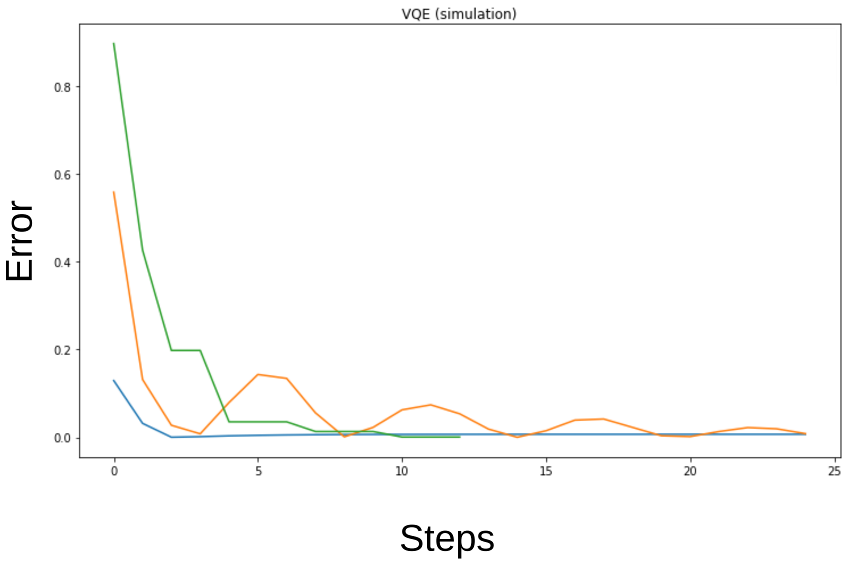


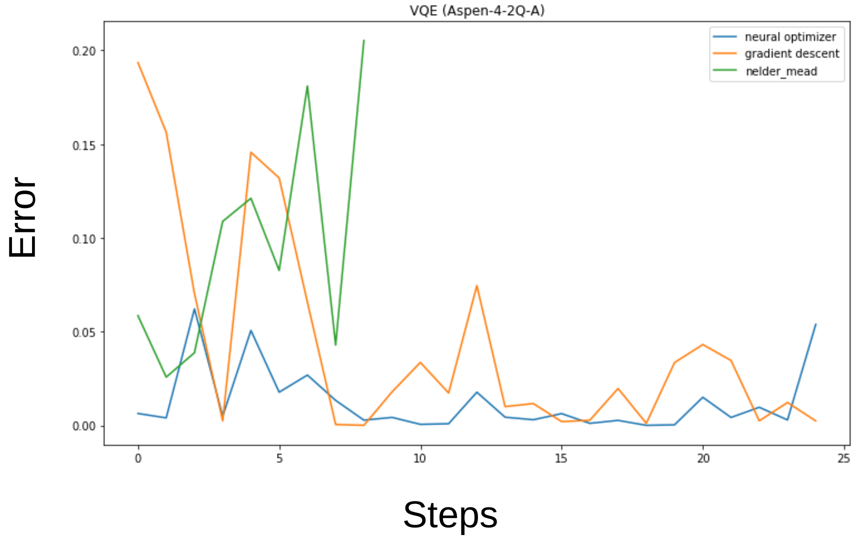
@kareldumon

MIRAEX

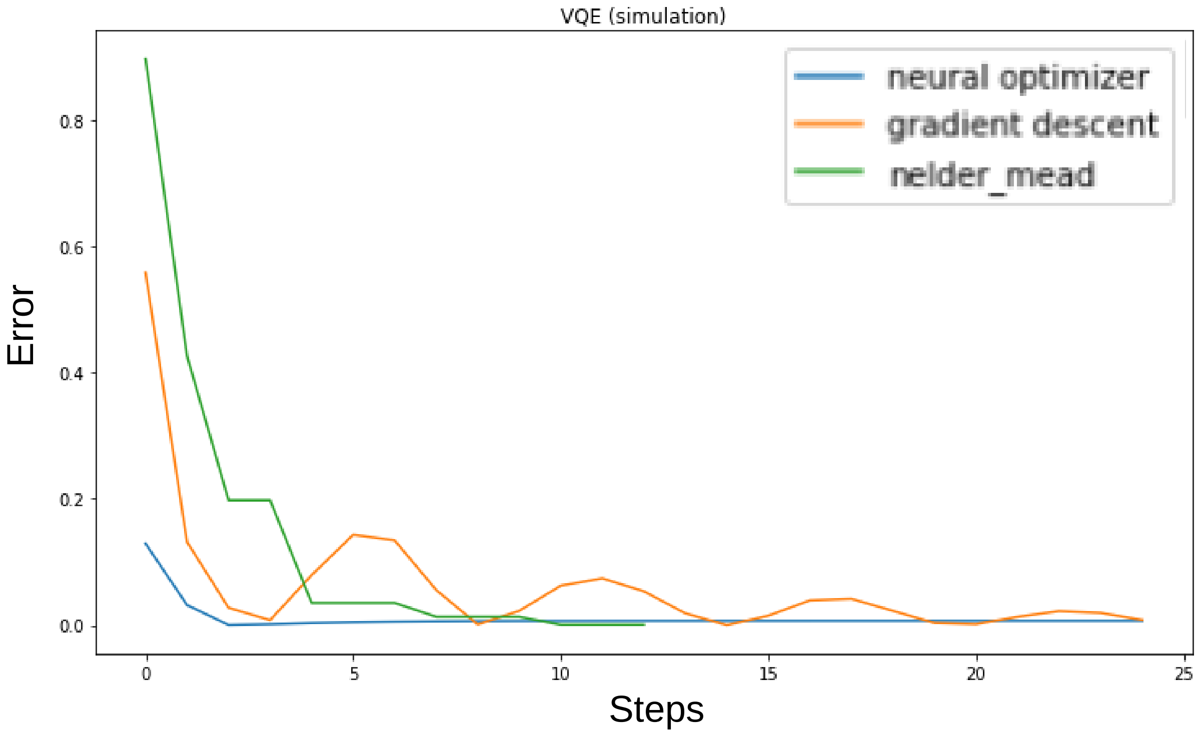
Example 2: VQE - Rigetti QPU







Example 2: VQE - IBM (simulation)



IBM Q

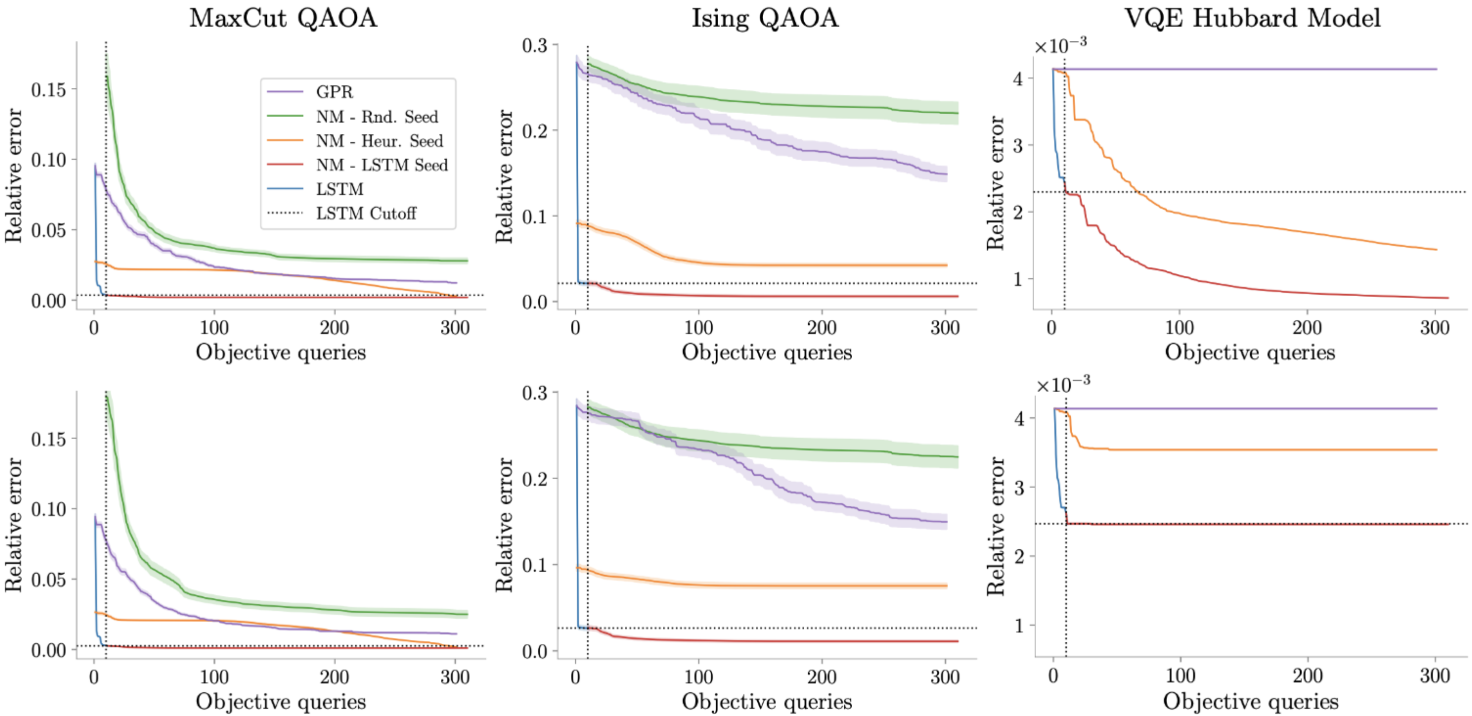
MIRAEX

@kareldumon

Faster (=cheaper) and scalable (= more qubits)

Add s

pape



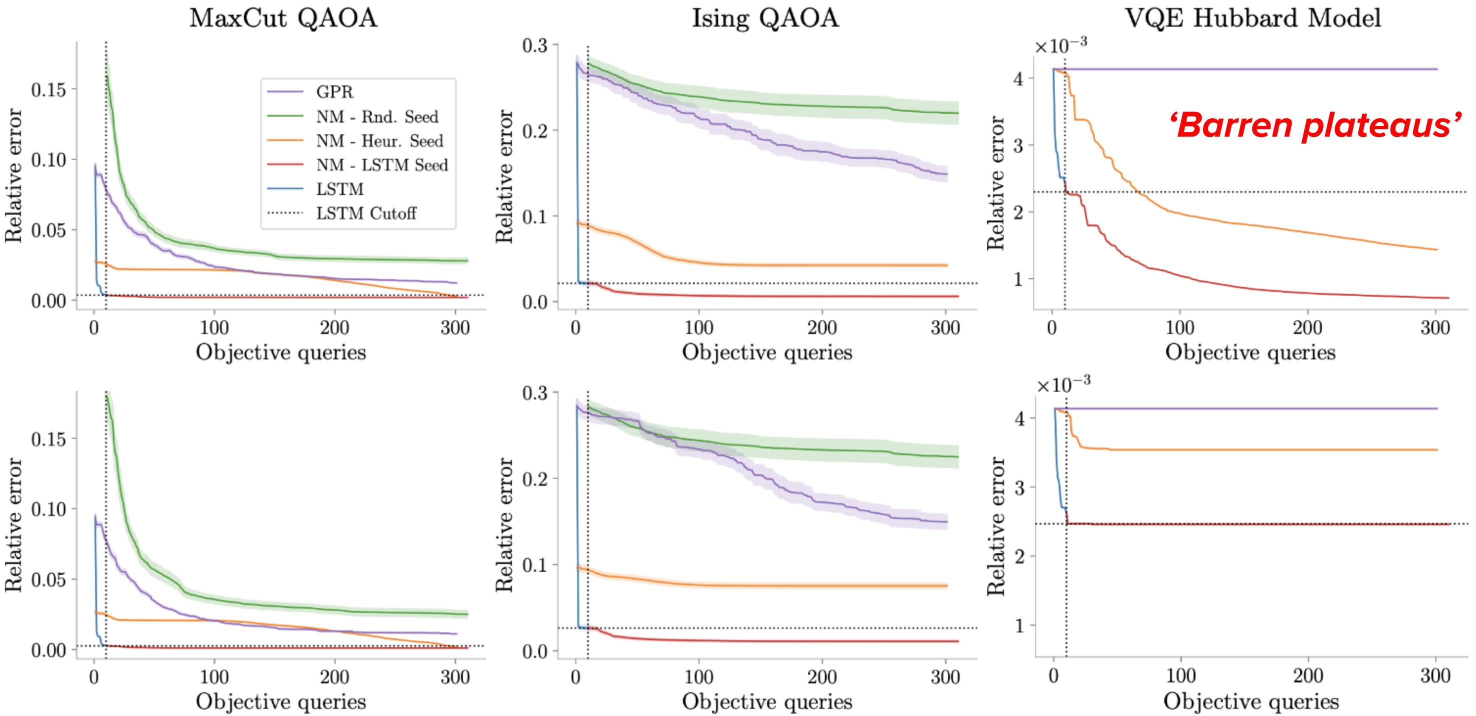
om

Faster optimization, also for bigger systems

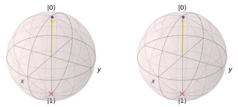
Add s

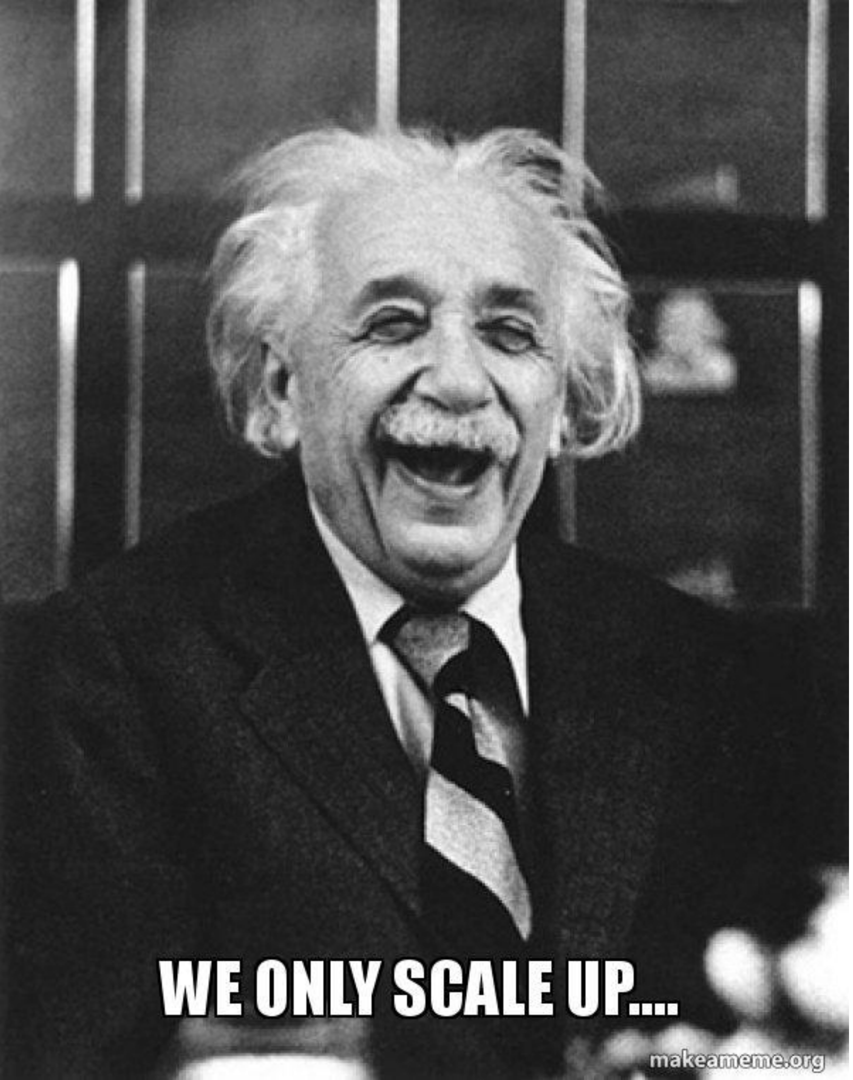
pape

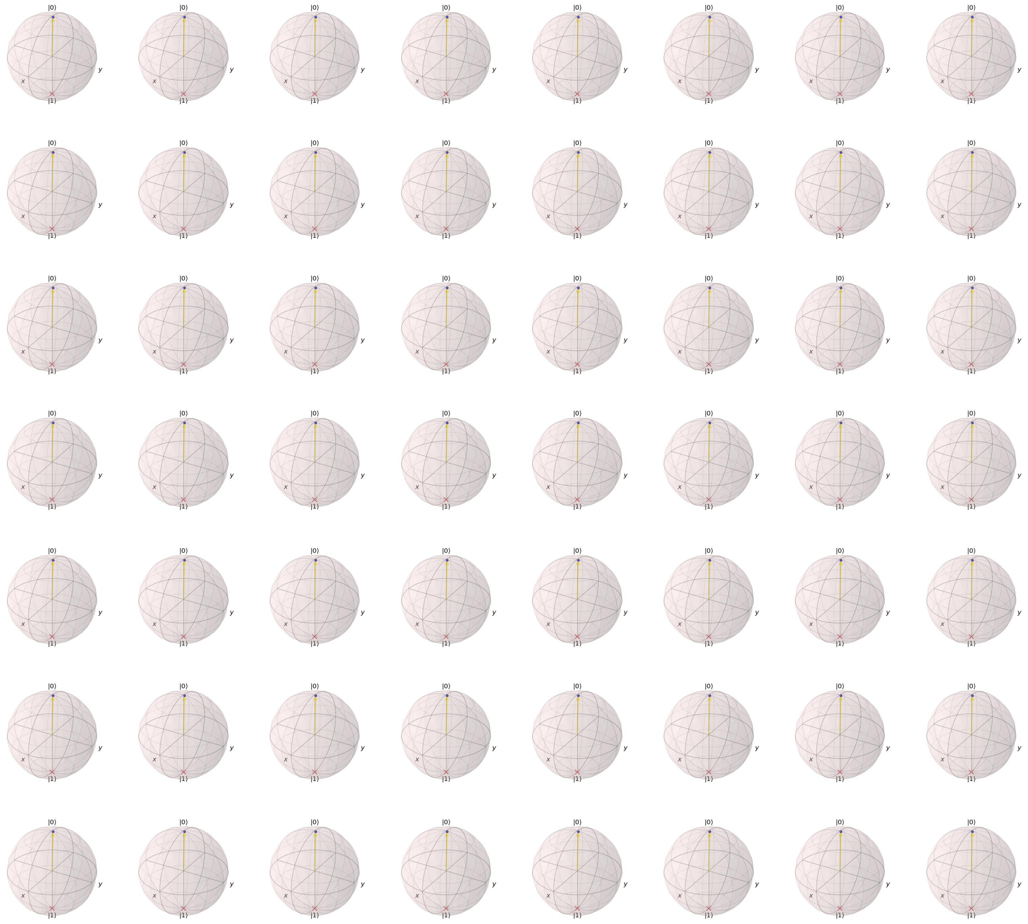
om



isn’t this too simple?

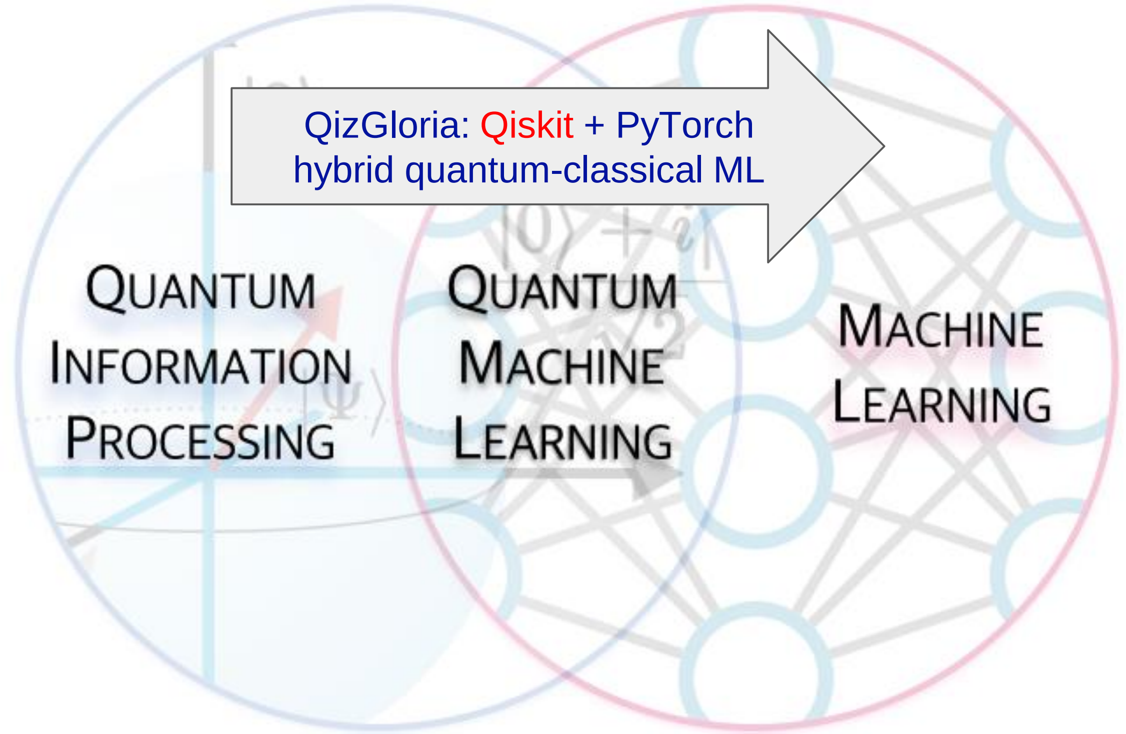



WE ONLY SCALE UP....
makeameme.org

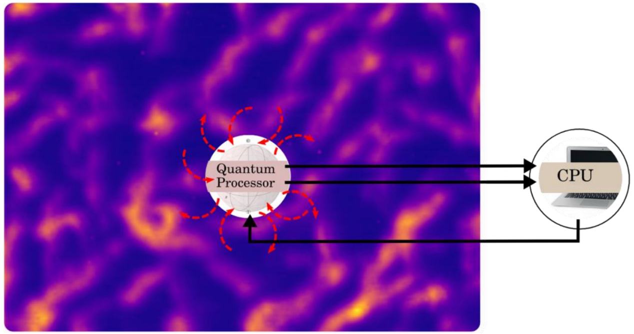


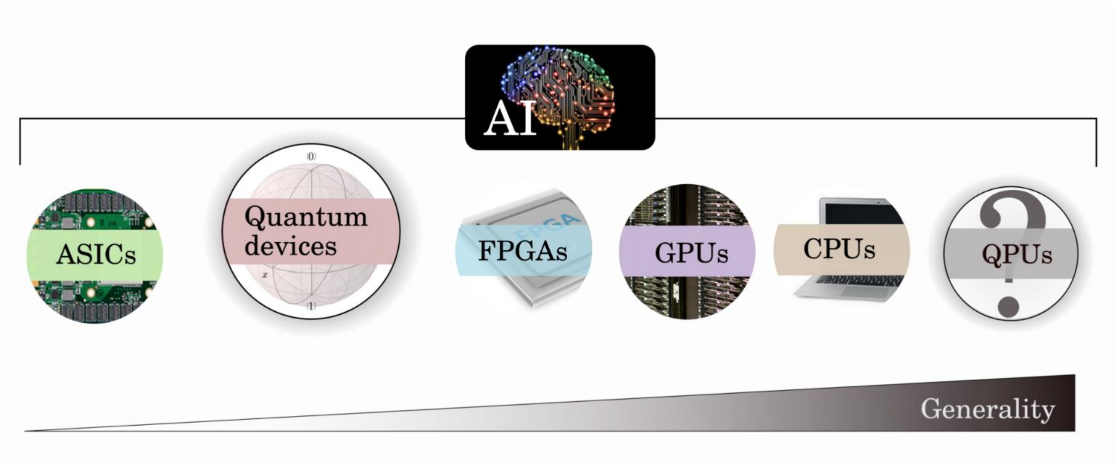
Quantum Neural Networks (QNNs) are still facing challenges...

but meta-learning can help!

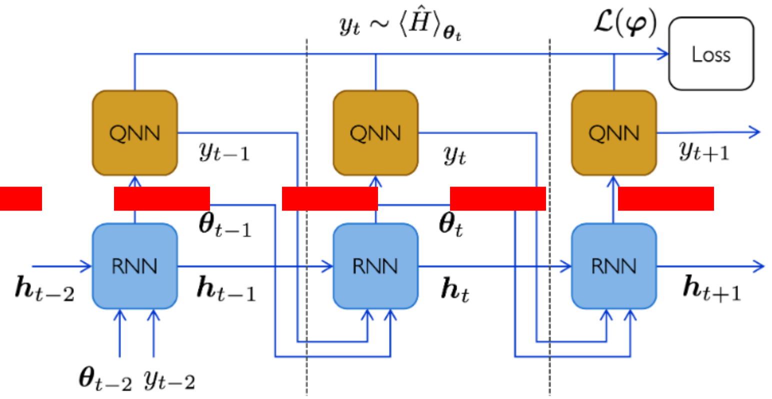

QizGloria: Qiskit + PyTorch
hybrid quantum-classical ML
QUANTUM INFORMATION PROCESSING
QUANTUM MACHINE LEARNING
MACHINE LEARNING

Exploiting NISQ hardware: Variational Circuits





Red line is hard boundary, that blocks us from tools



rigetti IBM Q

PENNYLANE PyTorch

XANADU


Aqua
Ignis
Aer
Qiskit
Terra
Hardware

Machine Learning with full PyTorch & Qiskit capabilities

IBM Qiskit Camp
Europe, 2019

Team QizGloria

Patrick Huembeli, Amira Abbas, Samuel Bosch,
Isaac Turteltaub, Karel Dumon
IBM Coach: Christa Zoufal

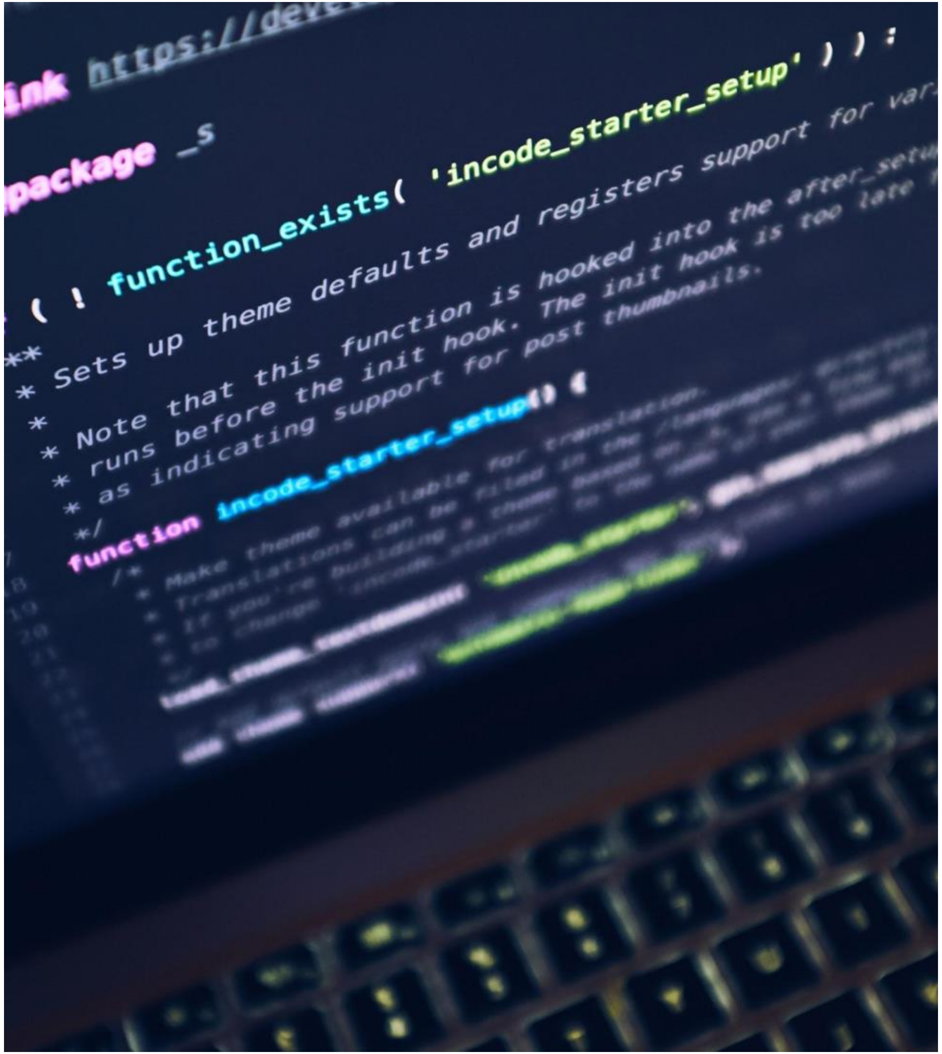
Our mission

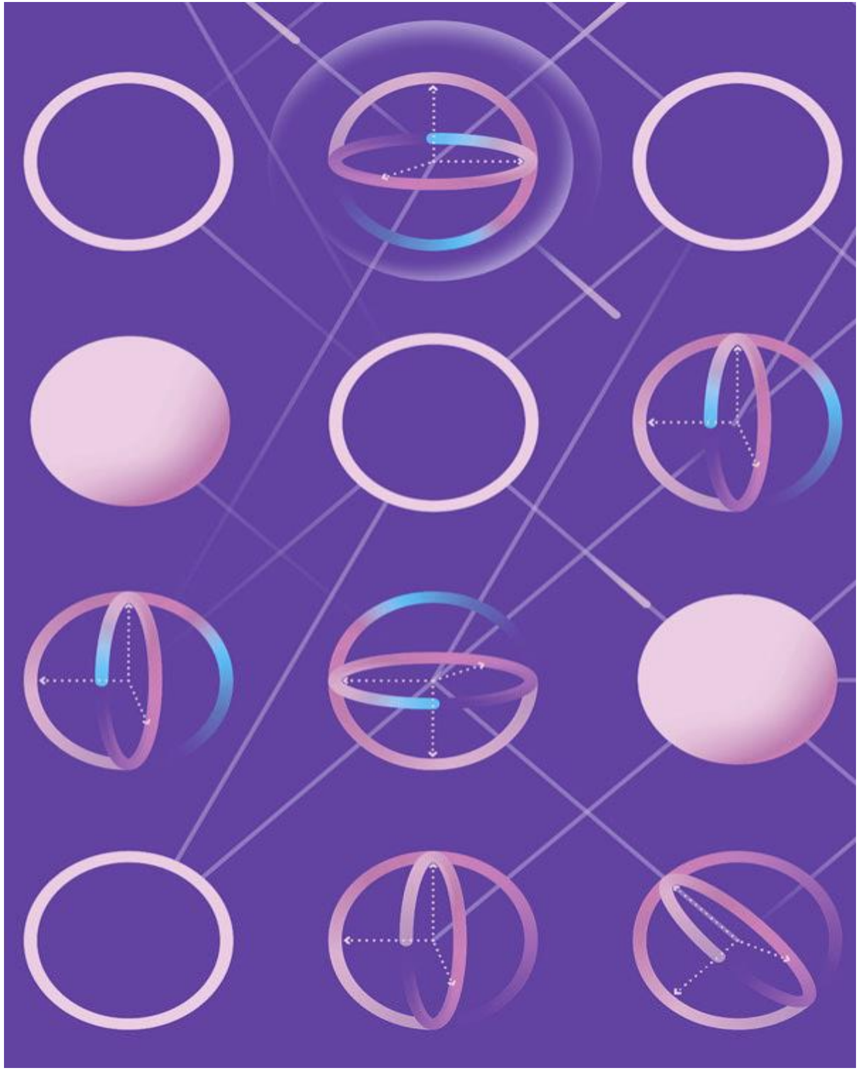
- Closer integration of Pytorch & Qiskit beyond existing tools

- Enable seamless co-training of quantum circuits & neural networks

- Encourage classical ML engineers to use quantum nodes

MIRAEX





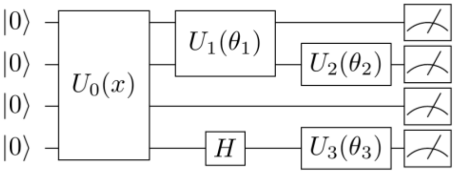
Why is this cool?

- PyTorch neural networks & tools
- More options for optimizers (RMSprop; Momentum; etc.)
- Full Qiskit capabilities (circuit definition, transpiler, Aqua,...)
- Back-end management by Qiskit (QPU, simulators)
- Bridges the gap between the QML and classical ML community

How does it work?

Qiskit

PyTorch



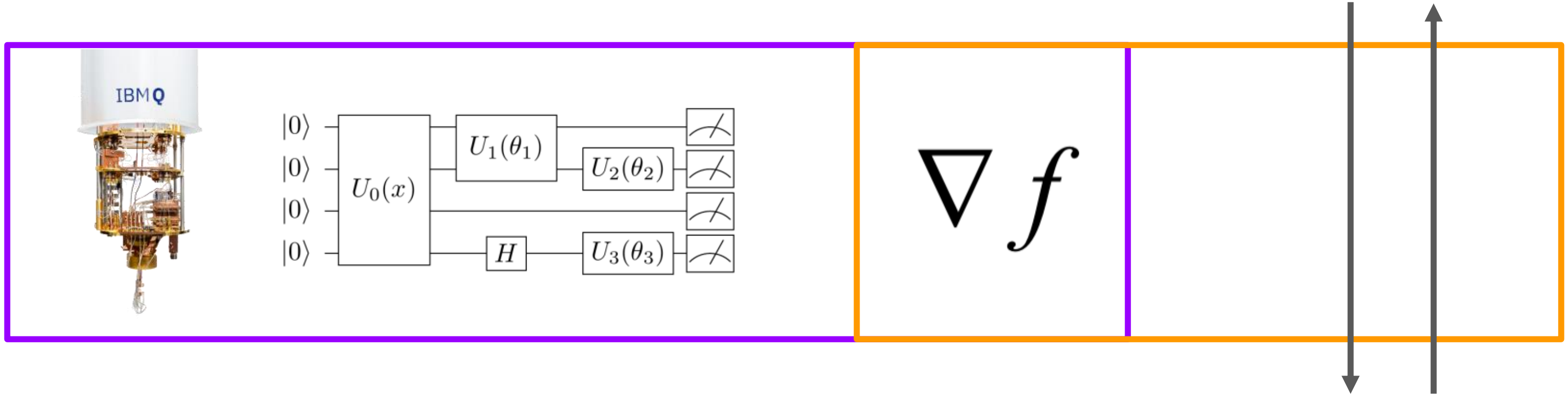
∇f

other frameworks:
blocked from Qiskit-tools!

How does it work?

Qiskit

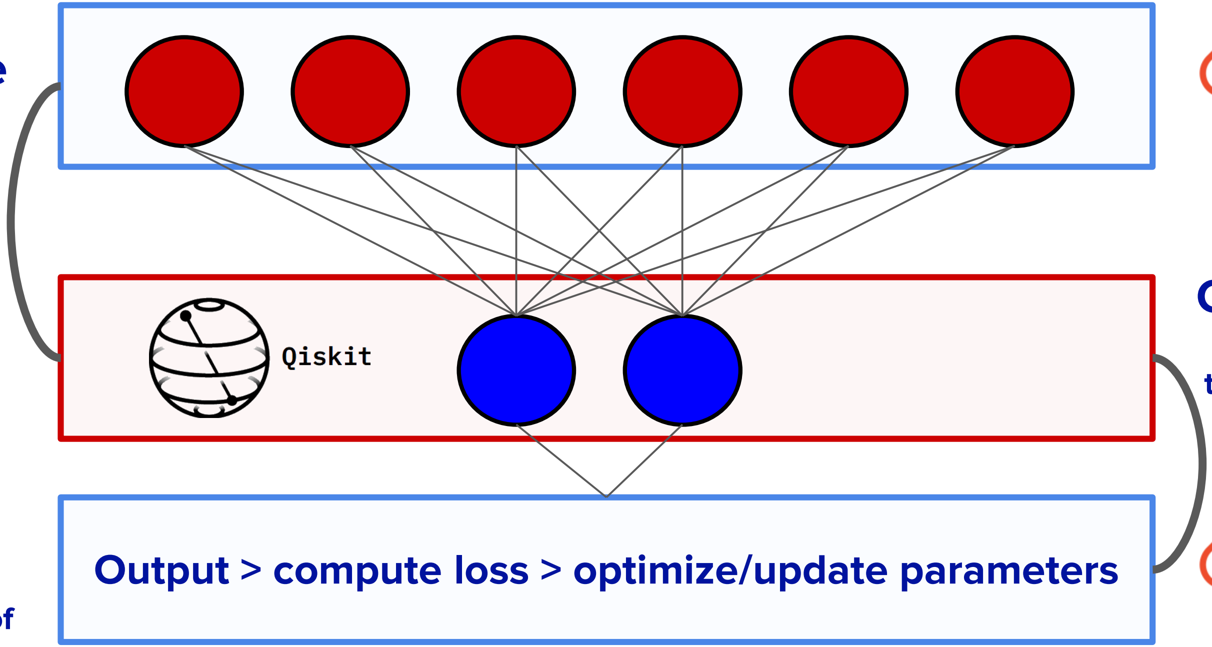
PyTorch



How does it work?

Classical node
(E.g. Pytorch neural network)

PyTorch



Quantum node
(Qiskit circuit with trainable parameters; VQE, QAOA etc.)

PyTorch

Pytorch Optimization
(Computes gradient of loss function wrt parameters)

How does it work?

Qiskit

PyTorch

QiskitCircuit

circuit definition (Terra, Aqua)
parameter binding
expectation value evaluation
back-end management

TorchCircuit

tensorization
parallelization
forward pass
backward pass (finite diff, aqgd)

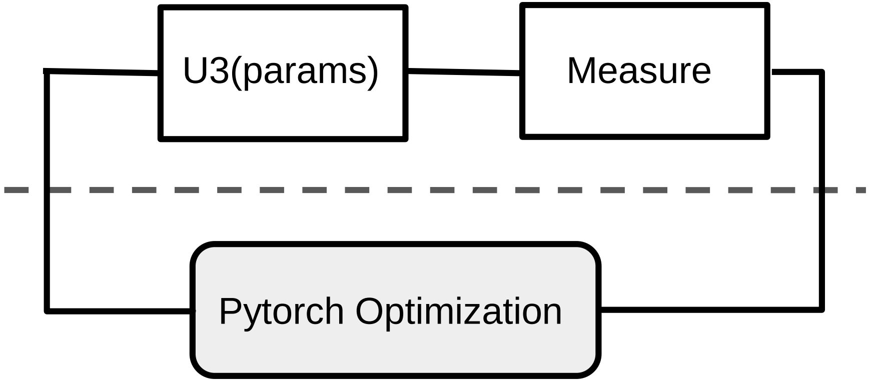
Seamless integration of Pytorch and Qiskit

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 10, kernel_size=5)
        self.conv2 = nn.Conv2d(10, 20, kernel_size=5)
        self.conv2_drop = nn.Dropout2d()
        self.fc1 = nn.Linear(320, 50)
        self.fc2 = nn.Linear(50, 3)

    def forward(self, x):
        x = F.relu(F.max_pool2d(self.conv1(x), 2))
        x = F.relu(F.max_pool2d(self.conv2_drop(self.conv2(x)), 2))
        x = x.view(-1, 320)
        x = F.relu(self.fc1(x))
        x = F.dropout(x, training=self.training)
        x = self.fc2(x)
        x = qc(x)
        return x
```

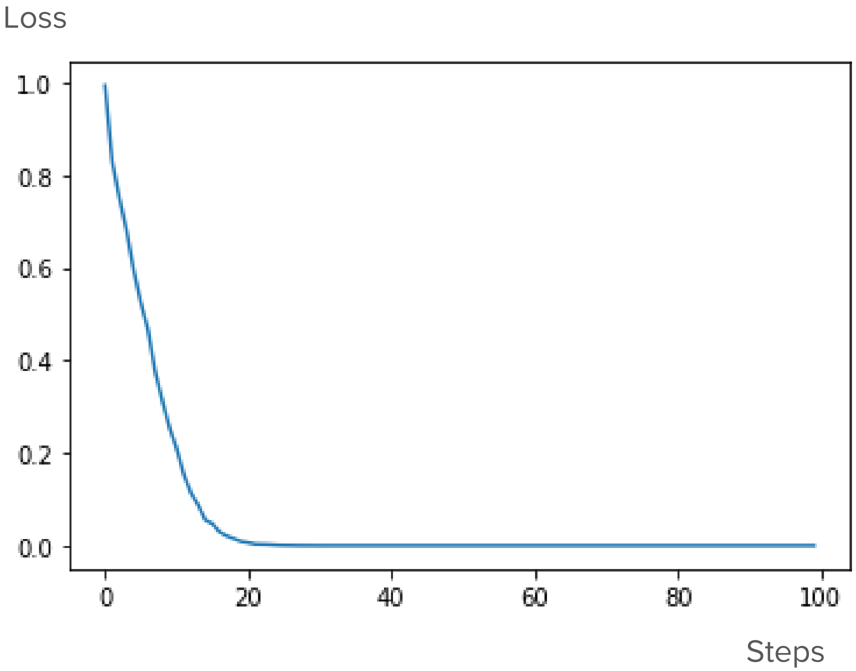
The options are endless with what you can do!

Hello World!

- Learn how to rotate 1 qubit to get a defined σ_z expectation
- Used U3 rotation in qiskit

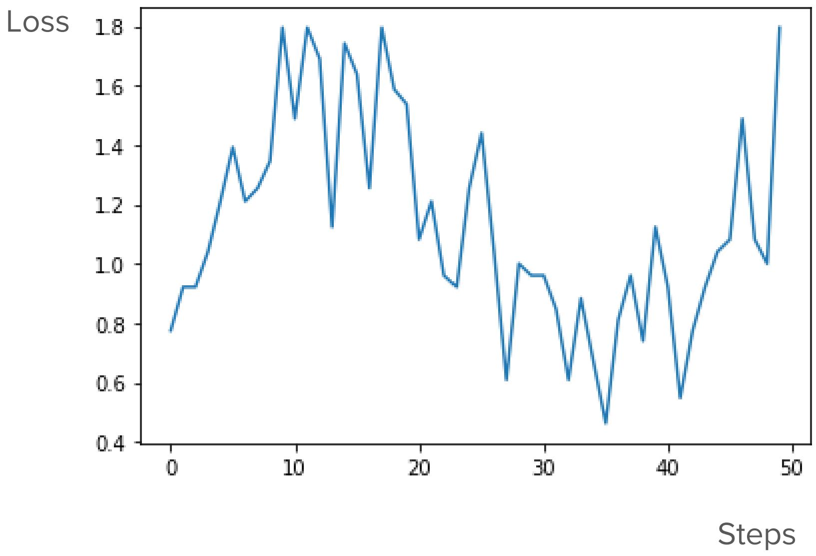


Update params

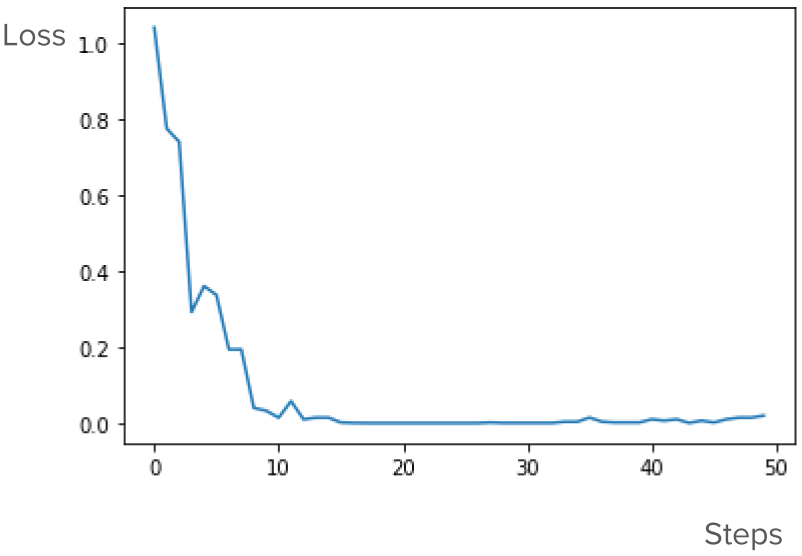


Details:
Finite difference gradient estimation; shots = 10 000

Hello World! - analytical gradients



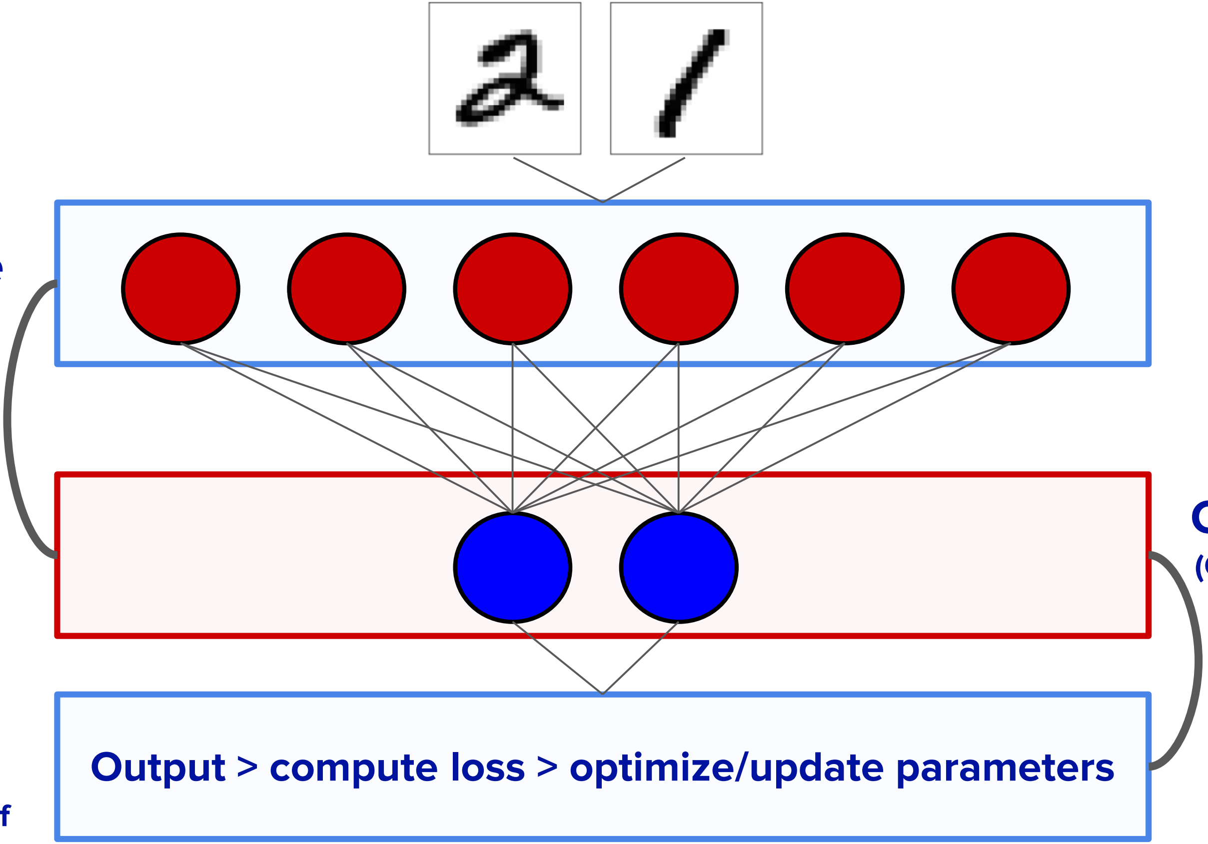
Details:
Finite difference gradient estimation;
shots = 100



Details:
Analytical gradient;
shots = 100

MNIST

Classical node ConvNet



Quantum node
(Qiskit circuit: Rx & Ry rotations)

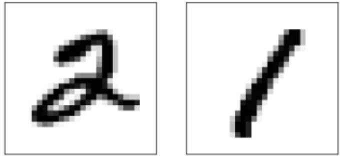
Pytorch Optimization
(Computes gradient of loss function wrt two parameters)

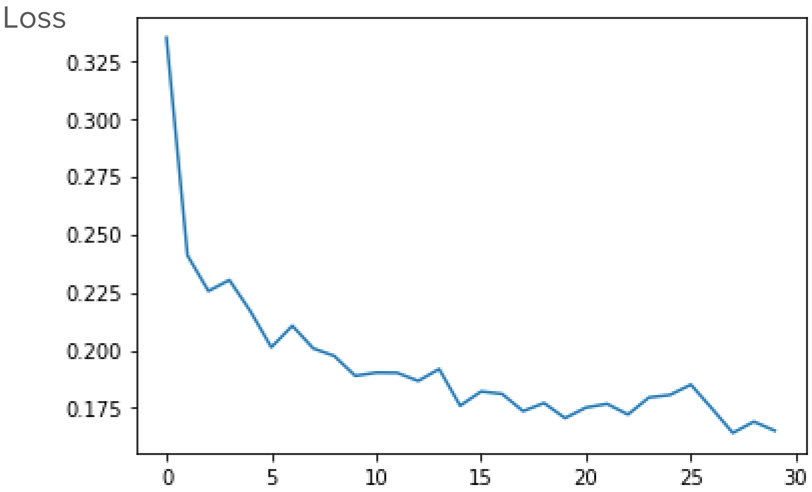
Sigma z exp > NLL/cross entropy loss > Pytorch Adam optimizer

MIRAEX

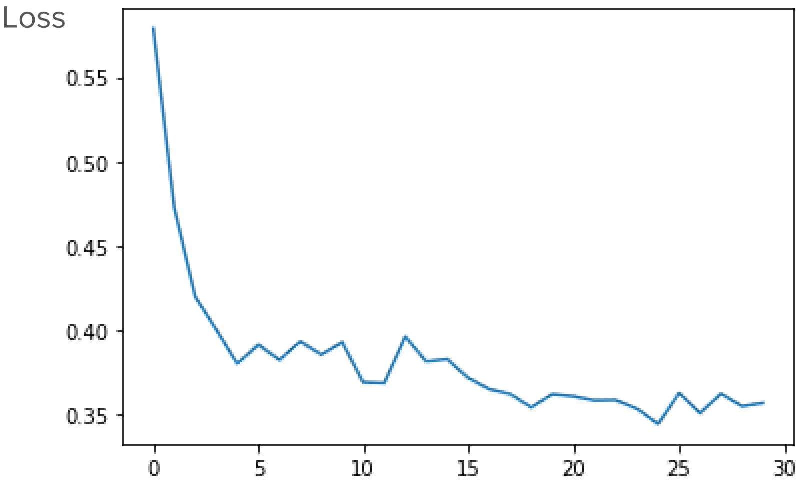
@kareldumon

MNIST





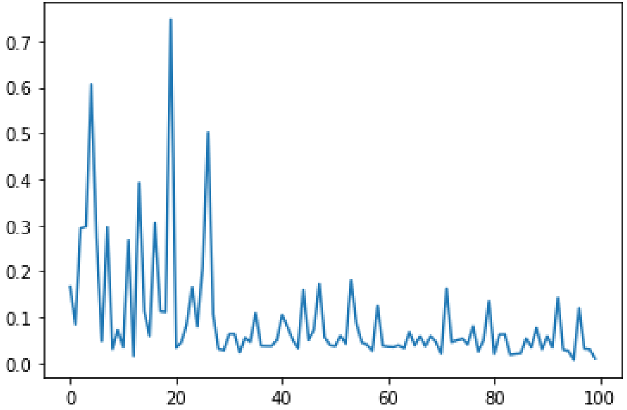
Details:
Analytical gradients
Negative-log-likelihood-loss
Shots = 100
(200 data samples per epoch)

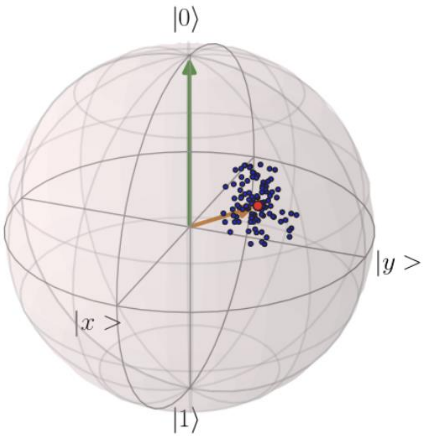


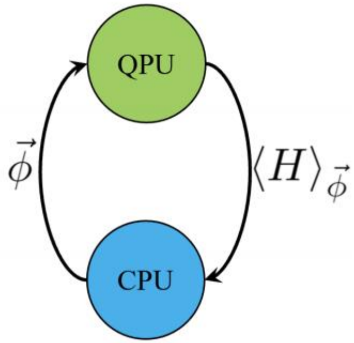
Details:
Finite difference gradient estimation
Cross-entropy loss for MNIST
Shots = 10 000

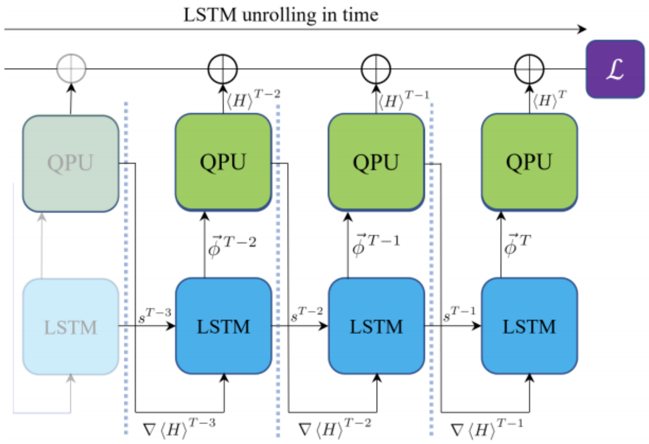
Other implementations we did

Meta-learning for neural optimizer for single qubit rotation









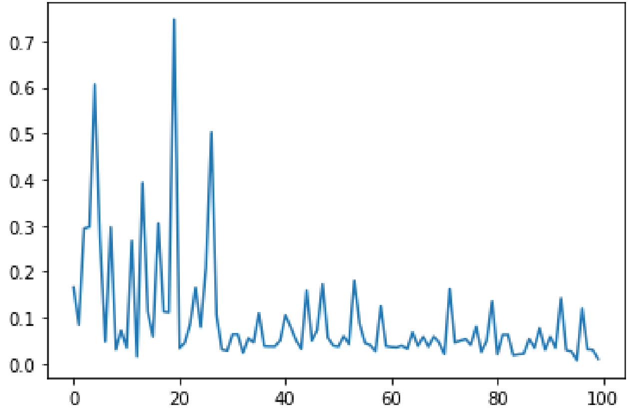
MIRAEX

@kareldumon

Wilson, Max, et al. "Optimizing quantum heuristics with meta-learning." *arXiv preprint arXiv:1908.03185* (2019).

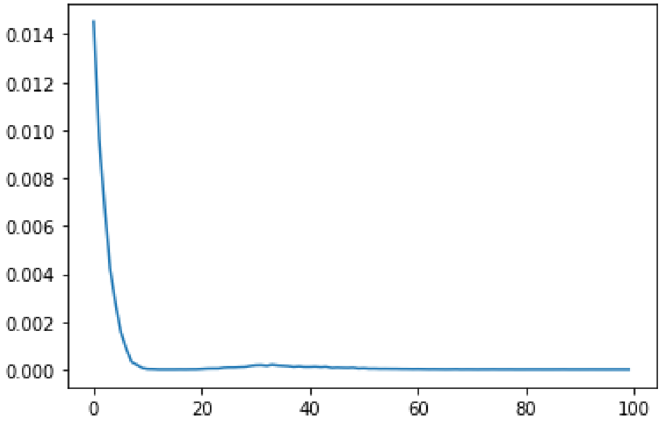
Other implementations we did

Meta-learning for neural optimizer for single qubit rotation

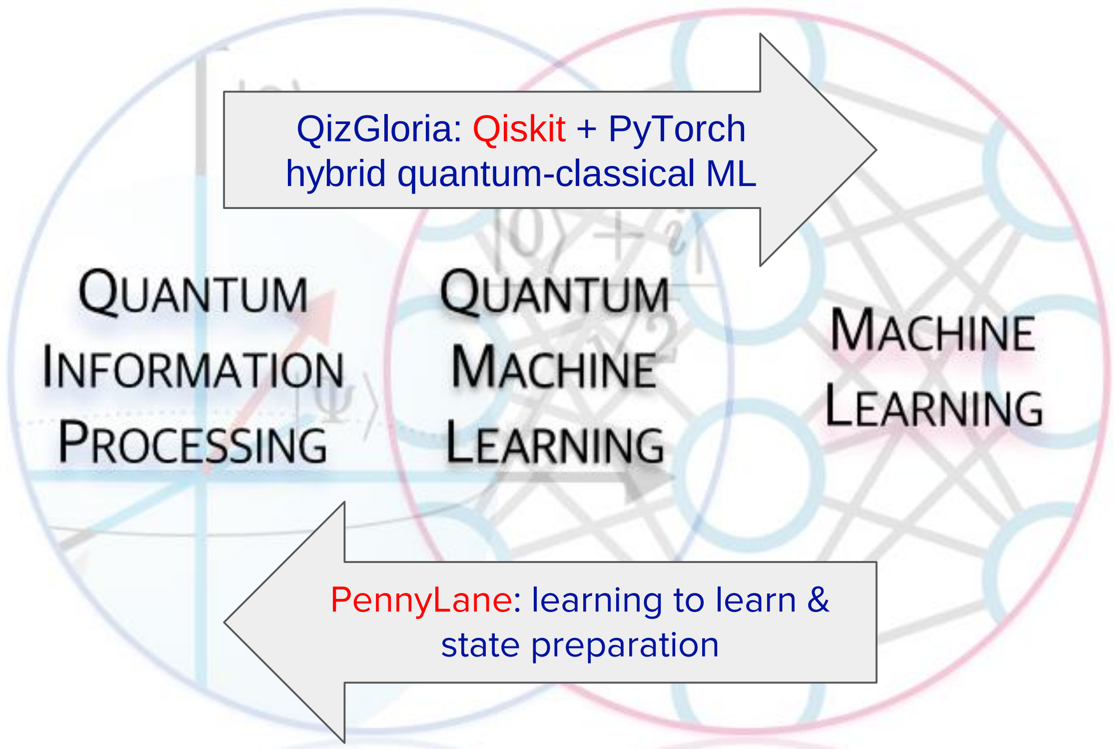


2 qubit example: Qiskit Aqua QAOA with Pytorch optimizers

- travelling salesman
- max-cut
- ...



Wilson, Max, et al. "Optimizing quantum heuristics with meta-learning." *arXiv preprint arXiv:1908.03185* (2019).



QizGloria: Qiskit + PyTorch
hybrid quantum-classical ML
QUANTUM
INFORMATION
PROCESSING
QUANTUM
MACHINE
LEARNING
MACHINE
LEARNING
PennyLane: learning to learn &
state preparation

distilling some

‘early development experiences/best practices’

Takeaway pe ruse case

- Different backends
- Open source tooling
- Collaborative mentality, also from the big players
- Idea sharing/hackathons
- Use cases/intros available online

NISQ is here! ... NISQ is here?

NISQ is the pre-'fault tolerant' era: qBLAS not possible!

- most people hear about applications for > 10^3 qubits, but we are not there yet
- so be suspicious: 'big data', 'exponential speedup', 'quantum data'...

NISQ: QPUs are climbing out of the lab

- but people are not sure how to make it useful
- personal view: a short burst of calculations in a great mathematical space

Potentially powerful device, still lots of room for creativity

- Similar approach as ML? (e.g. don't know why neural networks work)

Software can help quantum out of the lab

Classical machine learning can help improve quantum computing

- train in simulation (< supremacy), apply on real hardware (> supremacy) to save QPU time (i.e. money)

Software engineers: you can make an impact

- ‘physicists coding’: need software engineering tools, best practices & insights
- lots of ‘simple’ ideas still to be explored (cf. projects)
- it’s a great problem to work on
- lowering the barrier for other communities with similar problems/approaches, e.g. machine learning or software engineering (talent needed!)

‘Isolated’ full-stack models - which is best?

Each hardware & software framework has its own benefits...

- Rigetti: private QPU-time, co-processors, parametric compilation
- IBM Q: application focused, free access
- it’s an opportunity: winner is undecided & room for startups/good ideas

... but you don’t always have to choose from the beginning

- cf. PennyLane: flip different backends/simulations, helps break out of silos
- cross-pollination:
  - e.g. Qiskit 0.13 supports ion traps (hackathon result!)
  - e.g. Quantastica: Qiskit on Rigetti QPU
- room for co-development, e.g. partner models: cf. IBM Q, Rigetti, Xanadu,...

QM/QC looks daunting... but don't be afraid!

There are a lot of introductory tutorials

- From hardware providers: IBM Q experience / Qiskit textbook, Xanadu tutorials
- edX courses: quantum hardware / quantum machine learning
- qosf.org/learn_quantum/

The ecosystem is still small, and that's great!

- open-source mentality & room for co-development, cross-pollination
- IBM: Qiskit fully open source, partner model, events
- others (Rigetti, DWAVE,...): very eager for collaborations
- jump in Slack channels, join meetups, go to conferences,...

Start a new group | Explore | Messages | Notifications

Quantum Computing Belgium

Ghent, Belgium
212 members · Public group
Organized by **Karel Dumon**

Share:

About | Events | Members | Photos | Discussions | More

Manage group | Create event

What we're about

The Quantum Computing Belgium meetup hosts talks about the current state of technology in quantum computing and its implications for the future of...

Read more

Organizer

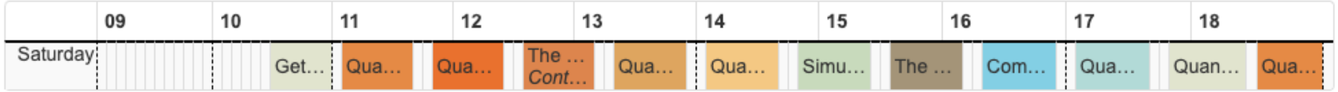
Karel Dumon
Message

Members (212)

See all

Quantum Computing devroom

Room: H.1301 (Cornil)
Calendar: iCal, xCal



Event	Speakers	Start	End
Saturday			
Getting started with quantum software development	Tomas Babej	10:30	11:00
Quantum machine learning with PennyLane	Joshua Izaac	11:05	11:40
Quantum computing hardware and control systems	Felix Tripier	11:50	12:25
The role of open source in building quantum computing ecosystem from scratch *Context of a developing country*	Hakob Avetisyan	12:35	13:10
Quantum Advantage and Quantum Computing in the Real World	Mark Mattingley-Scott	13:20	13:55
Quantum circuit optimisation, verification, and simulation with PyZX	John van de Wetering	14:05	14:40
SimulaQron - a simulator for developing quantum internet software	Axel Dahlberg	14:50	15:25
The Role of Open Source Frameworks in Quantum Computing and Technologies	Jack Hidary	15:35	16:10
Computing with TensorNetwork & QML Tools	Stefan Leichenauer	16:20	16:55
Quantum classifiers, robust data encodings, and software to implement them	Ryan LaRose	17:05	17:40
Quantum computer brands: connecting apples and oranges	Petar Korponaić	17:50	18:25
Quantum Open Source Foundation	Mark Fingerhuth	18:30	19:00

friendly reminder that you live in an age in which:

friendly reminder that you live in an age in which:

you can access a quantum computer...
on the other side of the world...
instantaneously...
for free...
from your sofa...

Just try it!

Quantum Development: Early Best Practices *from QML*

Innovation with Quantum Computing?!

GSE Architecture & Innovation Working Group

15 January 2020

Karel Dumon

Co-founder & Software Lead at Miraex

@kareldumon

MIRAEX

GUIDE SHARE EUROPE